MASTER'S THESIS

A study into Critical Success Factors during the adoption and implementation of Continuous Delivery & Continuous Deployment in a DevOps context

Vonk, R. (Robert)

Award date:
2020

A study into Critical Success Factors during the adoption and implementation of Continuous Delivery & Continuous Deployment in a DevOps context

Opleiding:	Open Universiteit, faculteit Management, Science & Technology Masteropleiding Business Process Management & IT
Degree programme:	Open University of the Netherlands, Faculty of Management, Science & Technology Business Process Management & IT master's programme
Course:	IM0602 BPMIT Graduation Assignment Preparation IM9806 Business Process Management and IT Graduation Assignment
Student:	Robert Vonk
Identification number:	
Date:	09-02-2020
Thesis supervisor	Prof. dr.ir. Jos J.M. Trienekens
Second reader	Michiel van Belzen MSc
Third assessor	N/a
Version number:	2.0
Status:	Final version

Abstract

Continuous practices research needs to increase in both number and, especially, rigour of empirical studies. Some studies conducted research into critical factors which apply to the adoption and implementation of Continuous Delivery and / or Continuous Deployment, but the definition of the critical factors mentioned in their studies was not based on scientific literature.

The objective of this research is to verify Critical Success Factors (CSF's) which apply to the adoption and implementation of Continuous practices in a DevOps context, that meet three criteria related to "CSF" scientific literature. This is done by carrying out a Systematic Literature Review (SLR) in order to find potential CSF's. After carrying out the SLR, 92 potential CSF's were found. These potential CSF's have been classified on the basis of similarity, which resulted in a newly compiled list of only 19 potential CSF's. Information about the three criteria in order to be a verified CSF have been retrieved by having interviews with DevOps team members.

All potential CSF's have been empirically verified, 6 potential CSF's contain performance improvements and no potential CSF's could be measured. No verified CSF's were found, which met all criteria in order to be a verified CSF.

These Critical Success Factors can make practitioners aware of the factors that may affect the success of Continuous Practices in their organizations. Further research is recommended in order to verify the remaining criteria on the list of potential CSF's.

Key terms

Continuous Integration, Continuous Delivery, Continuous Deployment, DevOps, Critical Success Factors, CSF's

Summary

Continuous practices research needs to increase in both number and, especially, rigour of empirical studies. Some studies conducted research into critical factors which apply to the adoption and implementation of Continuous Delivery and / or Continuous Deployment, but the definition of the critical factors mentioned in their studies was not based on “Critical Success Factors” scientific literature.

The main research question of this study is:

RQ: “Which Critical Success Factors apply to the adoption and implementation of Continuous Delivery and Continuous Deployment in a DevOps context?”

The objective of this research is to verify Critical Success Factors (CSF’s) which apply to the adoption and implementation of Continuous practices in a DevOps context, that meet three criteria related to “CSF” scientific literature.

This is done by carrying out a Systematic Literature Review (SLR) in order to find potential CSF’s. After carrying out the SLR, 92 potential CSF’s were found. These potential CSF’s have been classified on the basis of similarity, which resulted in a newly compiled list of only 19 potential CSF’s.

To answer the main research question, the list of 19 potential CSF’s was validated by empirical research on the following criteria:

1. Should have a verified significant impact on the success of Continuous Delivery or Continuous Deployment, which means the factor is empirical verified
2. Results in verified performance improvements
3. Success and performance may be measured

Some information about the three criteria in order to be a verified CSF, have been retrieved by having semi-structured interviews with members of different DevOps teams. These DevOps teams have successfully implemented Continuous practices in order to achieve their shared goals and / or performance requirements.

All potential CSF’s have been empirically verified, 6 potential CSF’s: *“Architecture”, “Resistance to change”, “Quality”, “Customer involvement”, “Test complexity & source code control”* and *“Pace”* results in verified performance improvements and no potential CSF’s could be measured. No verified CSF’s were found, which met all criteria in order to be a verified CSF.

The potential CSF's which have verified performance improvements may be carefully used when (new) DevOps teams want to adopt and implement Continuous practices, to indicate where the focus should be to improving the speed of delivery or achieving more story points per sprint. Further research is recommended to conduct this research in different case organizations, in order to verify the remaining criteria and to strengthen the validity of the list with potential CSF's.

Contents

1. Introduction

1.1 Background

This report is written as part of the Business Process Management and IT graduation assignment.
In this chapter attention is paid to background information and the aim of this research inside the topic of Continuous Delivery and Continuous Deployment. In the next chapter the definition of Continuous Delivery and Continuous Deployment and the related literature will be described in more detail. In the third chapter the research method is presented. The fourth chapter will describe the results of this study. Finally, in the last chapter a discussion, a conclusion and recommendations for practice and further research will be given.

1.2 Exploration of the topic

DevOps

Software-intensive companies constantly try to improve their software development process for better software quality and a faster time to market. (Lwakatare et al., 2016a) DevOps is an emerging paradigm to eliminate the split and barrier between developers and operations personnel that traditionally exists in many enterprises today. The main promise of DevOps is to enable continuous delivery of software in order to enable fast and frequent releases. (Wettinger et al., 2014) DevOps introduces a significant agile perspective to deliver the software product in short cycle time that will reduce technical debt that is caused by delay. (Toh et al., 2019)
Dyck, Penners, & Lichter (2015) proposed DevOps as an organizational approach stressing on good communication and empathy between teams, in addition they see continuous delivery as an established practice of release engineering. As a result, if an organization is implementing DevOps, release engineering becomes more effective, and a successful release engineering intensifies DevOps.

The study by Lwakatare, Kuvaja, & Oivo (2015) identifies four elements that characterize DevOps: collaboration, automation, measurement and monitoring. They also presented a conceptual framework to describe this DevOps phenomenon and also suggest that a DevOps implementation is necessary to enable Continuous Delivery. Lwakatare et al. (2016b) The leading paradigms of DevOps and Cloud computing help to implement comprehensive and fully automated deployment processes that aim to shorten release cycles by continuously delivering new iterations of an application. (Wettinger et al., 2015b) The advent of Cloud Computing has not only lowered the cost of IT operations but also enabled the notion of continuous delivery, which promises to radically reduce frictions in DevOps processes and speed up the product delivery cycle. (Austel et al., 2015)

In a similar vein, the recent emphasis on DevOps recognizes that the integration between software development and its operational deployment needs to be a continuous one. (Fitzgerald and Stol, 2014) Continuous Delivery and DevOps have emerged with the goal to bring together developers and operations personnel by enabling their efficient collaboration. This is technically supported by establishing automated continuous delivery pipelines to significantly shorten release cycles without quality degradation. (Wettinger et al., 2015a) The relationship between Continuous practices and DevOps according to Qumer Gill et al., (2017) implies that Continuous Delivery pipeline and continuous improvement are the most highlighted major DevOps processes.

This is also in line with Toh, Sahibuddin & Mahrin (2019), who are saying that Continuous Delivery is one of the DevOps‘ practices that enables software organization to release new features and new products rapidly. In addition, Greising, Bartel & Hagel (2018) are saying that a central aspect of DevOps is Continuous Delivery, i. e. the ability to release new software in an automated manner.

However, the DevOps phenomenon lacks clear definition and practices, and this makes it difficult for both researchers and practitioners to understand the phenomenon. (Lwakatare et al., 2016a)

In this study we adopt the definition of DevOps given by Erich (2019) as simply being "interaction between development and operations". This simple definition implies that DevOps is not a new thing which only certain organizations practice, but rather a fundamental characteristic of software and systems engineering that every organization is confronted with and manages to a certain extent.

Continuous Practices

Shahin, Ali Babar, & Zhu (2017) conducted a research aimed at systematically reviewing the state of the art of continuous practices to classify approaches and tools, identify challenges and practices in this regard. They have identified thirty approaches and associated tools, which facilitate the implementation of continuous practices. In their study they positioned Continuous Integration, Continuous Delivery and Continuous Deployment as examples of Continuous Practices. The aim of these Continuous Practices is to help organizations increase the speed between building and delivering new software features to a production environment without loss of quality. (J. Humble, and D. Farley, 2010).

Continuous Integration is defined as the part that takes care of compiling code, performing unit and acceptance testing, validating code coverage, checking coding standard compliance and building deployment packages. When performing Continuous Integration at a regular frequency, Continuous Integration should contribute to faster feedback to the developers, which can result in faster detection of integration errors and problems. (Fitzgerald & Stol, 2017) Continuous Integration is one of the main enablers of Continuous Delivery. (Rodriguez et al., 2017) This study focuses on Continuous Delivery and Continuous Deployment. Continuous Delivery ensures that all changes in an application are seriously tested and ready to be deployed. (I. Weber et al., 2016)

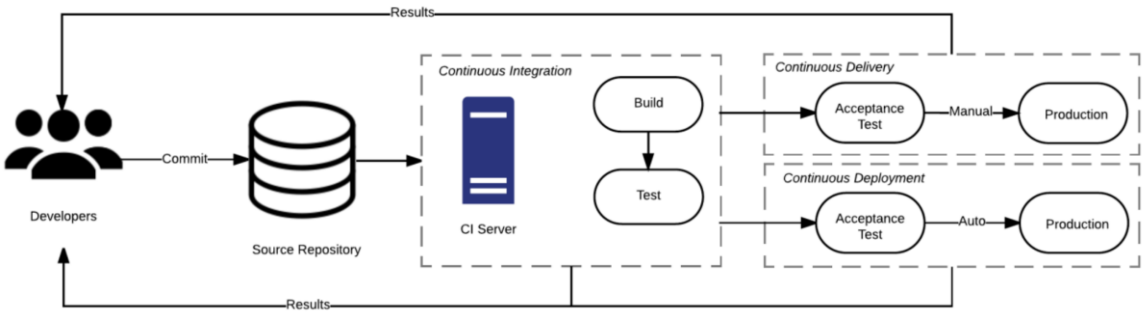

Figure 1 - The relationship between Continuous Integration, Continuous Delivery and Continuous Deployment (Shahin et al., 2017)

Continuous Delivery can be defined as a software engineering approach in which teams keep producing valuable software in short cycles and ensure that the software can be reliably released at any time. (L. Chen, 2015a) This practice seems to offer multiple benefits such as accelerated time to market, improved productivity and efficiency, reliable releases, improved product quality, building the right product and improved customer satisfaction. (L. Chen, 2017) According to Laukkanen et al. (2016) one simple reason why companies want to adopt Continuous Delivery is to shorten the software delivery cycle time. J. Humble, and D. Farley (2010) say 'Continuous delivery is more than just a new delivery methodology. It is a whole new paradigm for running a business that depends on software.'

In the study of Laukkanen, Itkonen, & Lassenius (2017), a systematic literature review was conducted to survey the faced problems when adopting Continuous Delivery. In addition, they identified causes for and solutions to the problems. They identified a total of 40 problems, 28 causal relationships and

29 solutions related to adoption of Continuous Delivery. The found problems, causes and solutions can be used to solve problems when adopting Continuous Delivery in practice.

Continuous Delivery aims to enable regular, rapid, reliable software releases through a set of sound practices, giving the people who “own” the software product the power to decide when to release changes. Continuous Delivery is a so-called pull-based approach, because software is “pulled” through the delivery mechanism when needed, and applies to all kinds of software (web, desktop, embedded, etc.). (Skelton & O'Dell, 2016)

Continuous Deployment goes a step further and automatically deploys those changes, as long as the quality gates are passed. (I. Weber et al., 2016) This is also in-line with Skelton and O'Dell (2016), they saying that Continuous Deployment is a push-based approach: when software developers commit a new feature to version control, it is pushed toward the Live systems automatically after successfully passing through a series of automated tests and checks. This definition of Continuous Deployment has also agreement with Laukkanen et al. (2017) who claimed that Continuous Deployment can be seen as an extension of Continuous Delivery. Continuous Deployment does not involve manual actions or decisions that need to be made to bring the new software features immediately to a production environment. This happens fully automatically with Continuous Deployment.

Rodriguez et al. (2017) conducted a systematic mapping study and classified the continuous deployment literature. The benefits and challenges related to continuous deployment were also analysed. An in-depth analysis of the primary studies revealed ten recurrent themes that characterize continuous deployment. However, there is a discussion in academic and industrial circles to reach consensus on the definition of continuous deployment and continuous delivery and making a distinction between the two terms. (Fitzgerald & Stol, 2017); (I. Weber et al., 2016);

In the study of Shahin, Ali Babar, & Zhu (2017) who have investigated what factors are important to successfully adopt and implement continuous practices, a list of critical factors for continuous practices success is established. They found 7 critical factors namely, “Testing (effort and time)”, “Team awareness and transparency”, “Good design principles”, “Customer”, “Highly skilled and motivated team”, “Application domain” and “Appropriate infrastructure”. These factors which may impact the success of continuous practices are defined as a critical success factor for making continuous practices successful.

Continuous Delivery adoption

According to figure 2, the focus area of this research are organizations that have not adopted Continuous Delivery and Continuous Deployment or are on their way to adopt Continuous Delivery and Continuous Deployment. Continuous Delivery adoption is defined as a set of actions a software development organization has to perform in order to adopt Continuous Delivery. (Laukkanen et al., 2017) Continuous Delivery adoption problems are problems that are directly preventing Continuous Delivery adoption and additional actions need to be done to solve them. (Laukkanen et al., 2017) In order to achieve a successful adoption of Continuous Delivery and Continuous Deployment it is necessary that there are no problems that are directly preventing Continuous Delivery adoption. Problems of Continuous Delivery are problems that emerge when Continuous Delivery is adopted. (Laukkanen et al., 2017) In order to achieve a successful implementation of Continuous Delivery and Continuous Deployment it is necessary that there are no problems that emerge when Continuous Delivery is adopted.

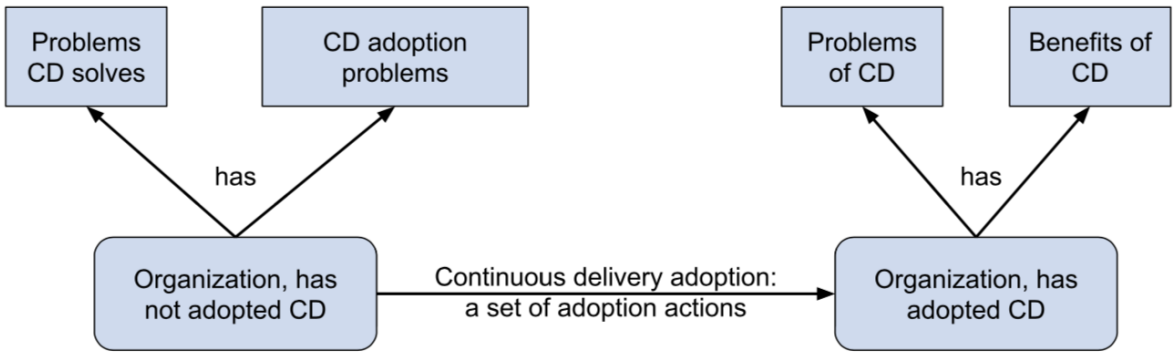

Figure 2 - Continuous Delivery adoption (Laukkanen et al., 2017)

Critical Success Factors

According to Leidecker & Bruno (1984) Critical Success Factors (CSF's) are those characteristics, conditions, or variables that when properly sustained, maintained, or managed can have a significant impact on the success of a firm competing in a particular industry.

In the study of Ram, Corkindale & Wu (2013) which investigated whether CSF's are in practice critical for achieving organisational performance improvements and the role that successful implementation may play in influencing the relationship between CSF's and improvements in organisational performance, they argue that a factor can only be termed a CSF if attending to this factor in a satisfactory manner results in performance improvements. Therefore, merely identifying a possibly important factor is not sufficient to constitute a CSF. The problem of establishing whether a CSF is really critical is further compounded by the multidimensional contexts in which 'success' and 'performance' may be measured, such as by user satisfaction or successful completion of a project, or through the tangible and intangible benefits to an organisation. Ram et al. (2013)

This problem is also confirmed by Markus et al. (2000) which argue that the definition and measurement of success are thorny matters. Success depends on the point of view from which you measure it. According to Esteves & Pastor (2000) this relative point of view for success can also be applied to failure, people will also qualify an implementation as a failure according to their goals. To accommodate the multidimensionality and relativity of enterprise system success from the adopting organization's perspective, Markus & Tanis (2000) define a standard of optimal success, which refers to the best outcomes the organization could achieve with enterprise systems, given its business situation, measured against a portfolio of project, early operational, and longer-term business results metrics.

Because Markus & Tanis introduce a manner to measure success by defining a standard of optimal success, in our view, we will use "the standard of optimal success" in the context of an organization which achieved the goal of Continuous Delivery and/or Continuous Deployment without any problems, in which the degree of success can be measured. The degree of success refers to the performance improvements which were caused by addressing these CSF's, measured against a portfolio of project, early operational, and longer-term business results metrics. (Markus & Tanis, 2000)

To measure performance improvements it is important to understand what exactly reflects the goal of Continuous Delivery and Continuous Deployment. According to Laukkanen et al. (2016) who are saying: "*One simple reason why companies want to adopt Continuous Delivery is to shorten the software delivery cycle time.* " which implies speed, is also confirmed by Rodríguez et al. (2017) which saying that "*Although similarities and differences exist among the emerging models of Continuous Deployment, three major themes characterize the models: (1) deployment, (2) continuity and (3) speed.*

Hence, continuous deployment means the ability to bring valuable product features to customers on demand and at will (deployment), in series or patterns with the aim of achieving continuous flow (continuity) and in significantly shorter cycles than traditional lead-times, from a couple of weeks, to days or even hours (speed)." In addition, Fitzgerald & Stol (2017) argue that "Achieving flow and continuity is much more important in first instance than speed."

In this study we define a CSF as a factor which:

1. Should have a verified significant impact on the success of Continuous Delivery or Continuous Deployment, which means the factor is empirical verified (Leidecker & Bruno, 1984)
2. Results in verified performance improvements (Ram et al., 2013)
3. Success and performance may be measured (Markus & Tanis, 2000)

A precondition is that the factor must have a clear description (citation), in order to assess the above criteria.

1.3 Problem statement

Research within the field of Continuous Delivery is still in its infancy and at an exploratory stage. A rigour and relevance analysis indicated that 37 primary studies exhibited high relevance; however, of these, only 14 studies showed high rigour and 23 studies had less than moderate rigour. This provides clear evidence that scientific contributions in the literature on Continuous Delivery are currently of high relevance but medium-low rigour, which calls for more meticulous research. Besides that, only twelve primary studies included a proper validity discussion (i.e. issues of bias, validity and reliability), whilst four primary studies mentioned validity, it was not described in detail. Thus, there is no description of threats to validity in 34 of the primary studies. Therefore, it can be concluded that there are important limitations to the scientific quality of the studies they retrieved and this inevitably reduces the reliability of the results. (Rodríguez et al., 2017)

With regard to the rigour in the systematic review study of Shahin, Ali Babar, & Zhu (2017) who established a list of critical factors (see chapter 1.2) for continuous practices success, they argue in their discussion that their systematic literature review has revealed the scarcity of reporting contextual information (e.g., organization size and domain) in the selected papers. To improve the quality and credibility of the results, researchers ought to report detailed contextual information. In addition, they derived their factors from their systematic literature review and consider them to be a critical factor for making continuous practices successful, when the factor is cited in at least 20% of the reviewed studies. This selection procedure to decide if a factor is a critical factor for Continuous practices success mentioned in their research, is not based on scientific literature about "Critical Success Factors". Because of this, it may be assumed that the quality is very low of the study with regard to the scientific rigour.

In addition, according to Markus and Tanis (2000) who conclude in their study that the definition of success depends on the point of view of the person who defines it. It is also unclear what exactly refers to Continuous practices success, it is unclear to what, the critical factors proposed by Shahin, Ali Babar, & Zhu (2017) are applying to. Besides that, according to the multivocal literature review study of Lwakatare et al. (2016b) they conclude that their results are limited (particularly in terms of the relationship of DevOps to agile, lean and Continuous Delivery practices). Most of the multivocal literature sources lacked empirical evidence or rigorous presentation of the authors opinions or experiences.

1.4 Research objective and questions

The aim of this research is to understand how to achieve a successful adoption and implementation of Continuous Delivery and Continuous Deployment in a DevOps context. By improving the understanding of CSF's in this study, an organization could use these CSF's in order to accelerate the delivery of software features.

The main question of this research (RQ) is: "Which Critical Success Factors apply to the adoption and implementation of Continuous Delivery and Continuous Deployment in a DevOps context?"

To answer the main research question, the research question is divided in the following subquestions:

SQ1: What is Continuous Delivery and Continuous Deployment?
SQ2: What are the corresponding CSF's?
SQ3: How are these CSF's addressed in a DevOps context?

1.5 Motivation/relevance

SQ1: What is Continuous Delivery and Continuous Deployment?

According to Rodríguez et al. (2017) there is no exact definition available in the literature of both concepts (Continuous Delivery and Continuous Deployment) and Continuous practices encompasses many different aspects. This research will make a contribution to the definitions of both concepts.
To identify which critical success factors apply to the success of the adoption and implementation of Continuous Delivery and Continuous Deployment it is important to clearly understand the definitions of these concepts and the aspects it contains. This is necessary in order to gain a deep understanding of the research domain.

SQ2: What are the corresponding CSF's?

According to Rodríguez et al. (2017) who conclude in their study that Continuous Delivery research needs to increase in both number and, especially, rigour of empirical studies. This research will make a contribution to the validation of CSF's, according to "CSF" scientific literature with use of criteria, to provide high quality in regard to the scientific rigour. According to Shahin, Ali Babar, & Zhu (2017) the critical factors can make practitioners aware of the factors that may affect the success of continuous practices in their organizations. In addition, according to Shahin, Ali Babar, & Zhu (2017) to improve the quality and credibility of the results, researchers ought to report detailed contextual information. This research will also take into account to report detailed contextual information.

SQ3: How are these CSF's addressed in a DevOps context?

According to Wettinger et al. (2015a) that Continuous Delivery and DevOps have emerged with the goal to bring together developers and operations personnel by enabling their efficient collaboration. It is asserted that "Collaboration" is one of the key dimensions of DevOps identified by Lwakatare, Kuvaja, & Oivo (2015). With regard to this element a DevOps context is chosen. This DevOps context is also chosen in response to the implications for further research proposed by Lwakatare et al. (2016b) with regard to the limitations that arise from the fact that a majority of sources are of low to medium pertinence in discussing the relationship of DevOps to agile, lean and Continuous Delivery practices. This research will make a contribution to the empirical validation of the found CSF's in a DevOps context in order to discuss the relationship of DevOps and Continuous Delivery practices.

2. Theoretical framework

This section provides the theoretical framework. Building the theoretical framework will be done by conducting a systematic literature review (SLR). According to Kitchenham (2004) a systematic literature review is a means of identifying, evaluating and interpreting all available research relevant to a particular research question, or topic area, or phenomenon of interest. Systematic reviews aim to present a fair evaluation of a research topic by using a trustworthy, rigorous, and auditable methodology.

2.1 Research approach

The rationale and question type for this research is to identify Critical Success Factors (CSF's) associated with the success of the adoption and implementation of Continuous Delivery and Continuous Deployment in a DevOps context.

In order to find the right primary studies a search strategy is conducted. (Kitchenham, 2004) The search strategy will be used in order to search for primary studies. The strategy contains search terms and sources which have to be searched, sources include databases, specific journals and conference papers / proceedings.

Within the available time for this study, it is not possible to analyse all of the available literature, therefore a choice was made to limit the number of digital libraries to five. IEEE, Scopus, Web of Science, ACM DL and SpringerLink libraries are used because these are major digital libraries in the scientific disciplines of this research (Computer Sciences and Software Engineering).

In order to formulate the search queries the sub questions of this study have been taken into account.

SQ1: What is Continuous Delivery and Continuous Deployment?

Objective: Gaining a deeper understanding of the concepts of Continuous Delivery and Continuous Deployment

In order to achieve this objective a search query is formulated. Because there is no exact definition of both concepts according to Rodríguez et al. (2017), the synonyms of Continuous Delivery and Continuous Deployment will be taken into account.

Because of the definition given by L. Chen: "*Continuous Delivery can be defined as a software engineering approach in which teams keep producing valuable software in short cycles and ensure that the software can be reliably released at any time*" (2015a, pp. 50) in which he indicates that Continuous Delivery is a software engineering approach, there is chosen to add "engineering" and "software engineering" as two extra related terms. In addition he also mentioned "released", which is a synonym for the delivery or deployment of software, because of this there is chosen to add "release" also as a synonym for "delivery" and "deployment".

According to Shahin et al. (2017b) who are saying: "*Continuous Delivery and Deployment (CD) practices aim to deliver software features more frequently and reliably.*" it can be deduced that the word "continuous" refers to a "frequent" cycle of the delivery of software features. This is why there is chosen to add the term "frequent" as a synonym for "continuous".

Besides that, according to Laukkanen et al. (2016) who are saying: "*One simple reason why companies want to adopt Continuous Delivery is to shorten the software delivery cycle time. "* it can be deduced that the reason to adopt Continuous Delivery is to shorten the software delivery cycle time, which implies increasing the "speed" of delivery or rather to provide e.g. "fast delivery of software" or "rapid

delivery of software". This is why there is chosen to add "rapid" and "fast" also as synonyms for "Continuous".

The keywords "adoption", "implementation" and "DevOps" derived from the research question are omitted, because this might be narrow the search results if it is used in combination with an AND operator, using these keywords in combination with an OR operator the results will possibly deviate too much from the topic.

Table 1 - Search term selection

Keywords:	*"Continuous Delivery" and "Continuous Deployment"*
Synonyms of "continuous":	*"rapid", "fast", "frequent"*
Synonyms of "delivery" and "deployment":	*"release"*
Related terms of "Continuous Delivery" and "Continuous Deployment":	*"engineering", "software engineering"*
Query with chosen combinations:	*("continuous delivery" OR "continuous deployment" OR "continuous release" OR "rapid delivery" OR "rapid deployment" OR "rapid release" OR "fast delivery" OR "fast deployment" OR "fast release" OR "frequent delivery" OR "frequent deployment" OR "frequent release" OR "delivery engineering" OR "deployment engineering" OR "release engineering" OR "continuous software engineering" OR "rapid software engineering" OR "fast software engineering" OR "frequent software engineering")*

The query according to table 1 will be used in order to broaden the area to search. The "OR" operator is used to broaden the results.

In order to search for primary studies, inclusion and exclusion criteria have been established.

To be included in the review, the study had to include the following:

1. Scholarly & peer-reviewed
2. Scientific articles, conference papers / proceedings and systematic literature reviews
3. Studies from 2001 - 2019
4. Written in english
5. Scientific discipline: Computer Sciences and Software Engineering
6. The abstract contains at least any of the keywords:
    - "continuous delivery"
    - "continuous deployment"
    - "continuous release"
    - "rapid delivery"
    - "rapid deployment"
    - "rapid release"
    - "fast delivery"
    - "fast deployment"
    - "fast release"

- “frequent delivery”
- “frequent deployment”
- “frequent release”
- “delivery engineering”
- “deployment engineering”
- “release engineering”
- “continuous software engineering”
- “rapid software engineering”
- “fast software engineering”
- "frequent software engineering”
- “release of new features”
- “release new features”
- “release of new software”
- “release new software”
- “releasing new software”
- “delivering new features”
- “deliver new features”
- “deliver new software”
- “deliver software”
- “deliver features”
- “delivery of new features”
- “delivery of software”
- “delivery of new software”
- “releasing new features”
- “deployment of new features”
- “deployment of software”
- “deployment of new software”
- “deployment of features”
- “deploy new features”
- “deploy software”
- “deploy new software”
- “deploy features”

7. Contain aspects like the definition, the goal, the characteristics or relationships with other practices of frequent/rapid or continuous delivery/deployment or releasing of software (features)

There is chosen to exclude studies that:

8. Do not contain an abstract
9. Contain duplicated articles
10. Are not being able to be imported/loaded to a reference manager
11. Are too technical and detailed
12. Do not have an available/retrievable .pdf version

Inclusion criteria number 3 is to focus the search for studies started from 2001, since then Agile Software Development has been introduced. (https://agilemanifesto.org/)

Inclusion criteria number 6 is used to cover a better focus area, there is chosen to add extra keyword combinations in comparison to the original search query. This is done especially because a lot of keywords e.g. “fast deployment”, “fast delivery” or “rapid delivery” possible not being mentioned in the abstract but the study is still focused on the deployment of software features (the keywords “delivery” and “deployment” are omitted in the search query because this resulted in a too wide search-area) This is why these extra keyword combinations are added.

Exclusion criteria number 8 is used to check the quality of the article. In this review only articles which contain an abstract are included. If an abstract is not available in the result of the digital library, Google (www.google.com) will be used to check if the abstract is available somewhere else on the internet.

Exclusion criteria number 10 is used in order to select articles which are capable of being imported/loaded to a reference manager. The reason this is required, is because in an SLR, it is easier to use a reference manager due the huge amount of articles for review to keep a clear overview. Some digital libraries does not provide options to export the results in BibTeX format which are compatible with “Zotero” (the reference manager used in this study) in order to import/load the results. In order to still import/load the results into the reference manager it is possible to do this in a different-way, namely by DOI (Digital Object Identifier). Zotero is able to import (bulk) DOI’s in order to save the results. So the articles which could not be exported in BibTex format are required to have a valid DOI. Articles which do not contain a DOI or the DOI is not found/reachable will be added manually. If all of the above options are not possible, the article will be excluded.

Exclusion criteria number 11 is used to assess the article on relevance to the subquestion, articles which are too technical and detailed do not provide information about aspects of frequent/rapid or continuous delivery/deployment or releasing of software (features), which can be used in order to formulate an appropriate answer to subquestion 1.

The SLR is divided into 6 steps, the results are presented in chapter 2.2. In the first step the search query from table 1 is used per digital library and inclusion criteria 1-5 and exclusion criteria 10 are being applied.

In the second step the results will be checked on the basis of duplicates of the articles. To execute this step, Zotero will be used in order to remove the duplicates of the found articles. Zotero identifies duplicates to compare all the meta-fields provided with an article. For this step exclusion criteria number 9 will be applied.

The third step consists of tracing the abstracts of the articles which did not have an abstract included in their export file. The abstracts of the articles will be searched in the selected digital libraries. When an abstract is found it will be copied into Zotero and linked to the designated article. This step is necessary because a lot of BibTeX / DOI imported articles come without the presence of an abstract. For this step exclusion criteria number 8 will be applied.

The fourth step will be a selection of articles based on a single search operation. For this step the reference manager "Zotero" will be used. This reference manager has the ability to apply multiple search criteria, to search in specific fields of the article in one single search operation. This fourth step is necessary in order to filter the articles which contain high relevance to sub question 1. In order to achieve high relevance, only articles who focus on frequent (rapid or continuous) release (delivery or deployment) of software (features) will be selected. The reference manager will search the abstract of the article. For this step inclusion criteria number 6 will be applied.

The fifth step is to make a selection of articles, based on reading the abstract. This step is necessary in order to provide articles which are being used for further analysis. The full article (.pdf version) of the selected articles will be downloaded. For this step exclusion criteria number 11 and 12 will be applied.

The final step is to make a selection of articles, based on reading the article. This step is necessary in order to provide articles that are being used to extract data from, in order to formulate an appropriate answer to sub question 1. For this step inclusion criteria number 7 will be applied.

Data extraction

If present, the following aspects will be extracted from the article:

- The meaning/definition of the practice (citation)
- The goal of the practice (citation)
- The characteristics of the practice (citation)
- Relationship of the practice with other practices (citation)

The full-article will be used in order to extract the above aspects. In case of multiple cases of the same aspect, but different citations of this aspect will be found in a single study, the citations will be seen as separate findings. The study will then be listed multiple times. In case of multiple times the same citation in a single study, the studies which support this citation, will be seen as a single finding and listed together.

Data synthesis

Because there is no exact definition of both concepts according to Rodríguez et al. (2017), first of all the found characteristics of Continuous Delivery and Continuous Deployment will be synthesized. The identified characteristics "name" and "description" will be used as categories in an Excel-sheet in order to classify the extracted meanings/definitions of the selected studies. All of the extracted meanings/definitions and the extracted goals of the different practices will be sorted per practice. After that, the citations from the meanings/definitions and from the goal are dissected into parts or words, that match the same word, share the same meaning or purpose as one of the categories (cross-checked). If they match a certain category the part or word(s) will be classified into that category. After all meanings/definitions and goals are classified, the results per category will be counted. It is assumed that a given definition or meaning always reflects the characteristics that are the most critical or important. The categories with the most hits, will be elected as the most critical or important characteristics. A new definition will be formulated for both concepts on the basis of the results of this classification. The most critical / important characteristics of both concepts will be merged in order to highlight the most critical / important characteristics in totality.
After that, the extracted citations about the relationship of the practice with other practices will be synthesized in order to describe the context of both concepts.

SQ2: What are the corresponding CSF's?

Objective: To establish an inventory of validated CSF's

In order to establish an inventory of validated CSF's the same query (table 1) and the same inclusion / exclusion criteria will be used, till the final step from SQ1. This consideration is because the fifth step of SQ1 is focussed on the area about frequent/rapid or continuous delivery/deployment or releasing of software (features) and has not yet divided into certain aspects. The results taken from the fifth step of SQ1 (N = 103), will be used as a starting point for SQ2.

Additional inclusion criteria has been established:

1. Contain aspects like problems, practices, factors, challenges, barriers, impediments, misfits, obstacles, failures, issues or lessons learnt of frequent/rapid or continuous delivery/deployment or releasing of software (features)
2. Contain (a) potential Critical Success Factor(s)

Inclusion criteria number 2 is used in order to identify potential CSF's. A precondition is that the potential CSF must have a clear description (citation), in order to assess the potential CSF on the CSF criteria established in chapter 1.2.

The first step is to make a selection of articles, based on reading the article. This step is necessary in order to provide articles that are being used to check the presence of potential CSF's. For this step inclusion criteria number 1 will be applied.

The second step is necessary to provide an inventory of potential CSF's in order to formulate an appropriate answer to sub question 2. For this step inclusion criteria number 2 will be applied. These articles will be used in order to extract data from.

Data extraction

- The aspect
- Name of the Critical Success Factor (citation)
- Description of the Critical Success Factor (citation)
- Information about the performance improvement or success (citation)
- Empirical evidence confirming the factor (citation)
- Information about how the CSF can be measured (citation)

Data synthesis

After all articles have been analyzed, the found potential CSF's will be merged in order to prevent duplicated CSF's. After that, the CSF's will be validated against the criteria defined in chapter 1.2, in order to be a CSF.

If a potential CSF contains data about all of the above criteria, which means:

1. Empirical evidence confirming the factor
2. Performance improvements due to the factor
3. The way the factor is measured

the CSF can be considered validated and included in the validated list of CSF's. The remaining potential CSF's will be included in the follow-up research in order to gain information about the missing criteria, so that these potential CSF's could also be assessed.

SQ3: How are these CSF's addressed in a DevOps context?

Objective: Collect as much information as possible about how these CSF's are addressed in a DevOps context

In order to give an answer on how these CSF's are addressed in a DevOps context, the same query (table 1) and the same inclusion / exclusion criteria will be used as far as the data synthesis from SQ2. This consideration is because the data synthesis of SQ2 provides an inventory of potential and possibly validated CSF's and for this question the target population are studies which contain potential or validated CSF's. The results taken from the data synthesis of SQ2 (N = 8), will be used as a starting point for SQ3.

Only one step is required in order to select the articles which contain potential or validated CSF's in a DevOps context. These articles will be used to search for information about the DevOps context and how the potential or validated CSF's were addressed.

Data extraction

- Information about the DevOps context (citation)
- Information on how the CSF was addressed in the DevOps context (citation)

Data synthesis

After all articles of the potential or validated CSF's are searched, the studies which provide solutions or information on how to address these potential or validated CSF's will be synthesized in a table. In addition, information about the contexts will be synthesized in a seperate table.

2.2 Implementation

SQ1: What is Continuous Delivery and Continuous Deployment?

The Systematic Literature Review (SLR) is divided into multiple steps.

Web of Science

Web of Science provided only the ability to search on "topic" which means according to Web of Science: "searches title, abstract, author keywords, and keywords plus" or search on "title". There was no option to search "only on abstract" directly. So there is chosen to search on "topic". One inclusion criterion could not be assessed completely, due to the limitation of the "custom year range" which was limited to 2009 and could not be set to 2001. Web of Science provided the possibility to download the results in BibTeX format including the abstract of the article. These results can be seen in Appendix H.

SpringerLink

SpringerLink provided only the ability to do a full-text search, there was no option to search "only on abstract" directly. So there is chosen to perform the full-text search. Both the articles and the conference papers of the SpringerLink library are included. However, the conference papers gave 1013 results. SpringerLink uses a maximum of 1000 results that can be made available for download. SpringerLink provided an export option as a .CSV (Comma Separated Values) output format only, which is not compatible with Zotero. Because of this limitation, the first 1000 results were downloaded in a .CSV (Comma Separated Values) file. The column "item DOI" in this file is used in order to perform the DOI (bulk) import to Zotero. Unfortunately there was no abstract included and 140 articles/conference papers did not contain a DOI in this .CSV file or were not available. These 140 results without DOI / not working DOI and the remaining 13 results which were not able to download, were manually imported to Zotero using the "Zotero browser connector". These results can be seen in Appendix F (articles) and in Appendix G (conference proceedings).

ACM DL

ACM DL provided multiple search options, e.g.(on title, on abstract or full-text). There is chosen to search on full-text. ACM DL provided the possibility to download the results in BibTeX format. Unfortunately there was no abstract included. These results can be seen in Appendix C.

Scopus

Scopus provided multiple search options, e.g.(on title, on abstract or on the basis of keywords). Because it was possible to search on abstract, there is chosen to perform to search on abstract directly. Because of this possibility, the inclusion/exclusion criteria of step 4 were taken into account and used as search query. Scopus provided the possibility to download the results in BibTeX format including the abstract of the article. These results can be seen in Appendix E.

IEEE

IEEE provided multiple search options, e.g.(on document title, on abstract or in document content). Because it was possible to search on abstract, there is chosen to perform to search on abstract directly. Because of this possibility, the inclusion/exclusion criteria of step 4 were taken into account and used as search query. Both the journals, magazines and conference publications of the IEEE digital library are included. IEEE provided an export option as a .CSV (Comma Separated Values) output format only, which is not compatible with Zotero. Results were downloaded in a .CSV (Comma Separated Values) file. The column "DOI" in this file is used in order to perform the DOI (bulk) import to Zotero.

Unfortunately there was no abstract included and 23 results in this .CSV file did not contain a DOI or were not available. These results were added manually by copying the BibTeX of the designated study (export options) into Zotero. These results can be seen in Appendix D.

- The results of the second step can be seen in Appendix I.
- The results of the third step can also be seen in Appendix I, the results which do not contain an abstract can be seen in the column "Abstract Note", which contain a blank field.
- The results of the fourth step can be seen in Appendix J.
- The results of the fifth step can be seen in Appendix K.
- The results of the final step can be seen in Appendix L.
- The extracted raw data can be seen in Appendix M.
- The results for subquestion 1 of this Systematic Literature Review (SLR) are synthesized per aspect which can be seen in Appendix A. The amount of found studies per aspect are drawn in chart 1 in Appendix AA.

The procedure followed can be seen in Figure 3 in Appendix AA for subquestion 1. Figure 3 in Appendix AA presents a visual representation of the taken steps and the resulting number of articles that were found after execution of each step.

SQ2: What are the corresponding CSF's?

The results taken from the fifth step of SQ1 (N = 103), will be used as a starting point for SQ2. The results of the fifth step can be seen in Appendix K.

- The results of the first step can be seen in Appendix N.
- The results of the second step can be seen in Appendix O.
- The results of the data extraction and the results of the data synthesis for sub question 2 of this Systematic Literature Review (SLR) can be seen in Appendix B. The amount of potential CSF's per aspect are drawn in chart 2 in Appendix AA.

The procedure followed can be seen in Figure 4 in Appendix AA for subquestion 2. Figure 4 in Appendix AA presents a visual representation of the taken steps and the resulting number of articles that were found after execution of each step.

SQ3: How are these CSF's addressed in a DevOps context?

The results taken from the data synthesis of SQ2 (N = 8), will be used as a starting point for SQ3. These results can be seen in Appendix O.

After completion of the last step, no articles were found in this Systematic Literature Review (SLR) which contain CSF's and were addressed in a DevOps context.

The procedure followed can be seen in Figure 5 in Appendix AA for subquestion 3. Figure 5 in Appendix AA presents a visual representation of the taken steps and the resulting number of articles that were found after execution of each step.

2.3 Results and conclusions

SQ1: What is Continuous Delivery and Continuous Deployment?

Definition

According to appendix A, the found definitions and goals of the concepts of Continuous Delivery and Continuous Deployment are very diverse. This confirms the statements found in the literature, for

example: "*Some previous authors have used the term continuous deployment when actually referring to continuous delivery (e.g. [3, 21]).*" (Laukkanen et al., 2016), "*Precise definitions of continuous delivery and deployment are often missing in both the literature and industrial circles [5, 7, 8].*" (Shahin et al., 2018) and "*There is robust debate in academic and industrial circles about defining and distinguishing between continuous deployment and continuous delivery [4, 19, 20].*" (Shahin et al., 2017)

According to Appendix A, the characteristics used in the definitions / meanings in the literature of both concepts are also widely divergent.

Characteristics

In order to understand what characterizes Continuous Delivery and Continuous Deployment, in the study of Rodríguez et al. (2017), they created a classification schema. They used the concept 'factor' to identify and categorize the recurrent themes in the literature related to Continuous Deployment. They identified factors defining aspects that, according to their reviewed primary studies, are relevant in achieving Continuous Deployment. That is, aspects that are frequently considered in the literature in order to implement Continuous Deployment in practice. (Rodríguez et al., 2017) As a result of their analysis Rodríguez et al. (2017) identified 10 recurrent themes (or factors) related to Continuous Deployment.

In this SLR the found factors identified by Rodríguez et al. (2017) are taken into account to establish a table with characteristics of Continuous Delivery and Continuous Deployment. Some studies confirm the factors identified by Rodríguez et al. (2017), other studies are used to extend the table with other characteristics. The identified characteristics of Continuous Delivery and Continuous Deployment from this SLR are described in table 2 in Appendix AA. Table 2 in Appendix AA contains the following column descriptions: The column "Name" represents a citation of the name of the characteristic as mentioned in the article. The column "Description" represents a citation of the description of the characteristic as mentioned in the article and the column "Study" represents the authors of the study which the characteristic is extracted from.

According to table 2 in Appendix AA, It seems that some characteristics are prerequisites to achieve Continuous Delivery and/or Continuous Deployment. These prerequisites have been derived by certain words in the description, which are showing that the characteristic is required to implement or contribute to enable Continuous Delivery and/or Continuous Deployment. These characteristics are marked "red" in table 2 in Appendix AA.

Some studies deviate from Rodríguez et al. (2017) point of view, like the study of Laukkanen et al. (2017) where the identified characteristics of Continuous Delivery are not completely in line with Rodríguez et al. (2017). Laukkanen et al. (2017) saying:

"To avoid stretching the concept of CD and keep our study focused, we use the definition in the book by Humble and Farley [1], which does not include all the factors identified by Rodríguez et al., (2017). For example, we investigate CD as a development practice where software is kept releasable, but not necessarily released frequently. In addition, our definition does not necessarily imply tight customer involvement or rapid experimentation. Instead, the focus of our study is on the continuous testing and quality assurance activities, especially automated activities. Our view is more properly described by the characteristics of release engineering as defined by Adams and McIntosh: branching and merging, building and testing, build system, infrastructure-as-code, deployment and release [15]."

The study of Schermann et al. (2016) confirmed the "Continuous testing" characteristic identified by Rodriguez et al. (2017). Schermann et al. (2016) saying: "*Live testing is used in the context of continuous*

delivery and deployment to test changes or new features in the production environment. This includes canary releases, dark launches, A/B tests, and gradual rollouts. Oftentimes, multiple of these live testing practices need to be combined (e.g., running an A/B test after a dark launch)."

Besides that, the study of Chen (2015b) confirmed two of the characteristics identified by Rodriguez et al. (2017) like "Fast and frequent release": "*Teams that practice CD ensure the software is releasable at any time. They usually make frequent releases, as frequent as multiple times* a day." (Chen, 2015b) and "Automation": "*To achieve reliable release, the deployment is usually automated. Before executing the production deployment, the deployment process and scripts have usually been exercised several times in different testing environments.*" (Chen, 2015b)

In addition, the study of Chen (2015b) identified also two other characteristics like "Delivering valuable software": "*The team continuously makes sure the software being developed provides value to customers. This implies the ability to quickly gather users' feedback on a feature once it is delivered. This also implies that a big feature will be decomposed into smaller ones, so that each one can be quickly delivered to get feedback early.*" (Chen, 2015b) and "Small size": "*The size of a user story is usually sufficiently small so that it can be finished within a week. The number of user stories in a single release is also small. Keeping such increments small helps to reduce the cycle time and release risks.*" (Chen, 2015b)

In addition, two other studies were found that zoom in on a certain characteristic of Continuous Delivery / Continuous Deployment. In the study of Karpanoja et al. (2016) "Responsibility" is identified: "*Based on our research, higher quality and other benefits of continuous delivery can be achieved by giving developers more responsibility. Still, the added responsibilities do not necessarily increase stress, but can actually decrease it, and at the same time improve motivation and job satisfaction.*" (Karpanoja et al., 2016) This characteristic is also supported by the study of Shahin et al. (2017b) which saying: "*Implementing CD practices demands skill-set and knowledge that are either brand new (e.g., tools for automating CD process), or lie at the intersection of development and operations responsibilities. Similar to [31], our study reveals that adopting CD broadens the scope of responsibilities and skill-set. It is evident that these changes are particularly significant for developers who sometimes take larger shares in operations activities [31]. Whilst developers are expected to take active part in deployment for successful adoption of CD, it should not demolish operations' functions. Broadening responsibilities of developers to a larger extent may negatively impact their productivity in core tasks. Shifting extensive amount of operations' responsibilities to developers could cause fear of losing jobs for Ops team and may negatively affect success of transition to CD. Organizations should extensively promote knowledge sharing among team members to complement areas of skill-set and collaboratively work towards a shared goal.*"

Difference between the concepts Continuous Delivery and Continuous Deployment

According to several studies Continuous Delivery is a prerequisite for Continuous Deployment. (Makinen et al., 2016); (Karpanoja et al., 2016) This is also in line with the viewpoint of Laukkanen et al. (2017) which saying that: "*Continuous deployment is an extension to Continuous Delivery in which each change is built, tested and deployed to production automatically [13]. Thus, in contrast to Continuous Delivery, there are no manual steps or decisions between a developer commit and a production deployment.*" This is also supported by the viewpoint of Itkonen et al. (2016) which is saying that: "*The Difference to Continuous Deployment is that production deployments are done according to separate decisions.*" which implies that manual decisions are necessary to deploy to production.

Based on the statements of Makinen et al. (2016); Karpanoja et al. (2016) and Laukkanen et al. (2017), in this study we agree with the viewpoint of Shahin et al. (2017) which saying:

"*What differentiates Continuous Deployment from Continuous Delivery is a production environment (i.e., actual customers): the goal of Continuous Deployment practice is to automatically and steadily deploy every change into the production environment. It is important to note that Continuous Deployment (CD) practice implies Continuous Delivery (CDE) practice but the converse is not true [20]. Whilst the final deployment in CDE is a manual step, there should be no manual steps in CD, in which as soon as developers commit a change, the change is deployed to production through a deployment pipeline. CDE practice is a pull-based approach for which a business decides what and when to deploy; CD practice is a push-based approach [23]. In other words, the scope of CDE does not include frequent and automated release, and CD is consequently a continuation of CDE. Whilst CDE practice can be applied for all types of systems and organizations, CD practice may only be suitable for certain types of organizations or systems [20, 23, 24].*"

Relationship with other practices - Continuous Integration

According to several studies there is a relationship between Continuous Integration and Continuous Delivery / Continuous Deployment. (Shahin et al., 2017); (Laukkanen et al., 2017)

Continuous Delivery extends Continuous Integration by continuously testing that the software is of production quality through an automated release process. (Itkonen et al., 2016); (Laukkanen et al., 2017)

In this study there is agreement with the definition / meaning of Continuous Integration found in the study of Shahin et al. (2017), which saying: "*Continuous Integration (CI) is a widely established development practice in software development industry [4], in which members of a team integrate and merge development work (e.g., code) frequently, for example multiple times per day. CI enables software companies to have shorter and frequent release cycle, improve software quality, and increase their teams' productivity [4].* ", "*This practice includes automated software building and testing [5].*" and "*Continuous Integration (CI) is the practice of integrating code changes constantly (i.e., at least daily) into the main branch and entails automated build and testing [4, 5].*"

According to the study of Greising et al. (2018) "*Continuous Delivery extends the process of Continuous Integration by automating steps such as acceptance and performance tests and transferring the new software version to the production environment where it is monitored before it is finally released [1].*"

According to the mentioned studies above and appendix A, in our point of view Continuous Delivery / Continuous Deployment extends Continuous Integration, because Continuous Integration contribute in order to merge/integrate code changes and uses Continuous Delivery / Continuous Deployment by automation of testing, debugging, building, and packaging software releases to expedite the development process by cutting down on manual and repetitive tasks. (Eddy et al., 2017); (Greising et al., 2018); (Shahin et al., 2017)

Relationship with other practices - DevOps

According to the study of Lwakatare et al. (2016b) which saying: "*The split between developers and operations is a major obstacle to fast and frequent software releases; DevOps can bridge the gap by enabling more efficient collaboration [44].*" it appears that the split between developers and operations is a major obstacle to fast and frequent software releases (one of the characteristics identified in this SLR of Continuous Delivery / Continuous Deployment).

The relationship between Continuous Delivery / Continuous Deployment and DevOps emphasizes the same goals and principles. (Karvonen et al., 2017)

This can be confirmed according to the study of Karvonen et al. (2017) which saying that, DevOps has its goal to bridge the gap between development and operations. (Karvonen et al., 2017); (Shahin et al., 2016) and Wettinger et al. (2015a), which saying that "*Continuous Delivery and DevOps have emerged with the goal to bring together developers and operations personnel by enabling their efficient collaboration. This is technically supported by establishing automated continuous delivery pipelines to significantly shorten release cycles without quality degradation.*"

In addition Makinen et al. (2016) also confirmed that "collaboration" is required for Continuous Deployment: "*Continuous deployment requires seamless collaboration between people working on various aspects of software engineering and a streamlined product flow from one stage to another. One development approach that relies on teams' ability to deploy features continuously in an automated manner, and is gaining increasing popularity, is DevOps [11,12] – a portmanteau of developers (Dev) and operations (Ops).*" (Makinen et al., 2016)

Finally, Wettinger et al. (2015a) indicates that: "*DevOps [4] is often considered in this context as an approach to improve the collaboration between development ('dev') and operations ('ops'). As a result of improved collaboration, new software releases can be made available much faster.*"

According to the identified relationships with other practices, in the mentioned studies above, it seems that only "collaboration" is another important characteristic of Continuous Delivery / Continuous Deployment and DevOps. Because of this finding, the characteristic is added to table 2.

Data synthesis

In line with Appendix A the results of the data synthesis are presented in chart 3 in Appendix AA and chart 4 in Appendix AA. To establish these charts, the extracted meanings/definitions and the goal of Continuous Delivery and Continuous Deployment were cross-checked against the found characteristics of table 2. Chart 5 in Appendix AA describes the results of "what" Continuous Delivery and Continuous Deployment (chart 6 in Appendix AA) stands for.

Conclusion SQ1

According to Rodríguez et al. (2017) there is no exact definition available in the literature of both concepts (Continuous Delivery and Continuous Deployment). This is also confirmed by Shahin et al. (2017c). According to our quantitative approach the following definitions would be established:

Chart 3 in Appendix AA shows that "*Fast and frequent release*", "*Continuous testing and quality assurance*" and "*Automation*" clearly are the most important characteristics of Continuous Delivery, if we apply a rule which consider a minimum of 5 hits to be "important". Derived from these findings and derived from the descriptions of these characteristics from table 2, Continuous Delivery could be considered as a "*software engineering discipline*" (according to chart 5 in Appendix AA) which stands for: "*The ability to release software whenever the organization wants to (on demand or at will) based on need and with preference given to shorter cycles or even continuous flow (weekly or daily), ensuring the quality of the software at all times and automating the delivery pipeline from building and testing to deployment and monitoring.*"

Chart 4 in Appendix AA shows that "*Fast and frequent release*", "*Continuous testing and quality assurance*", "*Automation*" and "*Customer involvement*" clearly are the most important characteristics of Continuous Deployment, if we apply a rule which consider a minimum of 5 hits to be "important". Derived from these findings and derived from the descriptions of these characteristics from table 2, Continuous Deployment could be considered as a "practice" (according to chart 6 in Appendix AA) which stands for: "*The ability to release software whenever the organization wants to (on demand or at will) based on need and with preference given to shorter cycles or even continuous flow (weekly or daily), ensuring the quality of the software at all times, automating the delivery pipeline from building*

and testing to deployment and monitoring and having mechanisms to involve customers in the development process and collect customer feedback from deliveries as early as possible (even near real-time) to drive design decisions and innovation."

Rodríguez et al. (2017) are saying: "*A general consensus exists among most authors of the literature surveyed that Continuous Deployment refers to the ability of organizations to release software functionalities to customers quickly and frequently, soon after each new functionality is developed.*"

In our view this definition seems to be referring to the characteristic of "*Fast and frequent release*", because of the word "frequently", the characteristic "*Automation*" by using the word "quickly" which we assume this means: "the fully automated delivery pipeline what enables the speed of the delivery process" and the characteristic "*Customer involvement*" by saying "to release software functionalities to customers". In comparison to our quantitative approach, which uses the citations of meaning/definitions (Appendix A) of Continuous Delivery and Continuous Deployment to establish a new definition for both concepts, we fully agree with the definition given by Rodríguez et al. (2017), because chart 4 is showing that those three characteristics are having a high score in relation to the total.

In contrast to Chen (2018) who is saying that "*Continuous Delivery (CD) is a software engineering approach in which teams keep producing valuable software in short cycles and ensure that the software can be reliably released at any time.*"

In our view this definition seems to be referring to the characteristic of "*Fast and frequent release*", because of the word "short cycles", the characteristic "*Continuous testing and quality assurance*" by using the sentence "*ensure that the software can be reliably released*" which we assume this means: "the software (feature) is free of bugs and the developers feel confident to release the software (feature)" which implies "quality". In our opinion, "at any time" refers more to a business decision which has to be made and do not implies "speed or quick delivery" by automation of the delivery pipeline. This means that the characteristic of "*Automation*" will be missing. This conclusion is also supported by Rodríguez et al. (2017) which are saying that: "*In addition, some authors distinguish between delivery and deployment. For example, Humble and Farley (2010) describe Continuous Deployment as the automatic deployment of every change to production, whilst Continuous Delivery is an organizational capability that ensures that every change can be deployed to production, if it is targeted (however, the organization may choose not to do it, usually due to business reasons). However, most of the scientific literature uses the terms Continuous Deployment and Continuous Delivery interchangeably."* For example: *"Continuous Delivery (also known as Continuous Deployment [1]) focuses on the automation of the delivery process of software."* (Greising et al., 2018)

In comparison to our quantitative approach, this definition for Continuous Delivery given by Chen (2018) is also supported because of the high scores on the characteristics of "*Fast and frequent release*" and "*Continuous testing and quality assurance*" in relation to the total and the lower score of the characteristic "*Automation*" in relation to the score of that characteristic for Continuous Deployment, in consideration of the unequal amount of gathered meanings/definitions per concept.

SQ2: What are the corresponding CSF's?

Data synthesis

The data synthesis concerning the complete list of the potential CSF's including the validation according to the criteria defined in chapter 2.1 of sub question 2 can be found in Appendix B. These potential CSF's (name and description) are assessed by a cross-check against the characteristics (name and description) which are a prerequisite of Continuous Delivery and / or Continuous Deployment

according to table 2 in Appendix AA in order to check the completeness of the found CSF's and to consider them as CSF's which are prerequisites. This cross-check can be seen in Appendix P.

Conclusion SQ2

A total of 92 potential Critical Success Factors were found of which 34 potential Critical Success Factors seem to be prerequisites for Continuous Delivery and / or Continuous Deployment. The list of potential CSF's is established in table 3 in Appendix AA. No completely validated CSF's were found, because there were no manners found which measure the CSF's and only a few performance improvements were found.

Table 3 in Appendix AA contains the following column descriptions: The column "Potential CSF Name" represents a citation of the name of the factor as mentioned in the article. The column "Description" represents a citation of the description of the factor as mentioned in the article and the column "Study" represents the authors of the study which the potential CSF is extracted from. The column "Pre" refers to the potential CSF's who seem to be prerequisites for Continuous Delivery and / or Continuous Deployment, these are indicated by a red marking.

SQ3: How are these CSF's addressed in a DevOps context?

Conclusion SQ3

No CSF's addressed in a DevOps context were found. The reason for this, is because information about the context is not mentioned in the selected articles. It is unclear if the potential CSF's are related to a certain context. This is also confirmed by Shahin, Ali Babar, & Zhu (2017), their systematic literature review has revealed the scarcity of reporting contextual information (e.g., organization size and domain) in the Continuous practices domain.

Final conclusion

By carrying out the Systematic Literature Review, answers can only be given to sub questions 1 and 2. New definitions were drawn up for both concepts based on the most important / critical characteristics derived from the found definitions in the literature. In addition, an inventory was drawn up of potential CSF's that contribute to the successful adoption and implementation of Continuous Delivery and Continuous Deployment. Unfortunately, it was not possible to answer the question of how these found CSF's are addressed in a DevOps context.

2.4 Objective of the follow-up research

Referring to sub question 3 which remained unanswered, it is important to know how these CSF's have been addressed in a DevOps context and subsequently led to a successful adoption and implementation of Continuous Delivery and Continuous Deployment. In addition, a lot of information about the criteria defined in chapter 1.2 is missing about the potential CSF's. It is necessary to retrieve this extra information during the follow-up research.

The objective of the follow-up research is to validate the list of potential CSF's in a DevOps context. In this way, it can be examined whether these CSF's are actually recognized in an organization and can be confirmed with practical examples, measurement reports or documents and to find out which CSF results in what performance improvements or success.

3. Methodology

3.1 Conceptual design: select the research method(s)

The objective of this part of the research is to verify the list of validated CSF's empirically in a DevOps context. In this way, it can be examined whether these CSF's are actually recognized in an organization and can be confirmed with practical examples, measurement reports or documents. The reason to carry out this research is to provide an answer to the main research question: RQ1: "Which Critical Success Factors apply to the adoption and implementation of Continuous Delivery and Continuous Deployment in a DevOps context?"

To achieve this objective, the type of information required to answer this is evidence of found and possible additional CSF's in terms of practical examples, (measurement) reports or documents. Information which indicates that the CSF is recognized, will verify the CSF.

This type of information can be found by experts who are working at an organisation within a DevOps team and which have implemented Continuous Delivery and/or Continuous Deployment. The reason there is chosen for experts which already take part of the Continuous Delivery and/or Continuous Deployment process, is because it is examined whether these CSF's played a role during the adoption and implementation and how the organization have addressed these CSF's. This is only possible when the implementation phase is over.

There are mono or multiple methods to provide this information like a quantitative method, a qualitative method or a mixed methods research design. (Saunders et al., 2016) In a quantitative way, for example, surveys could be conducted among the experts in a DevOps team of the organization and the CSF's could be statistically analysed. This could also be done in a qualitative way by conducting semi-structured interviews and in-depth information about the CSF's could be retrieved.

For this research design there is chosen for a (mono) qualitative method in order to deliver this information. The established list of CSF's is in line with a deductive approach, to test an existing theory using qualitative procedures. (Yin, 2014) The reason there is chosen for a qualitative method is because it is an explorative case study (Saunders et al., 2016) and as much in-depth information as possible needs to be retrieved. Deskresearch can be a good first step. Existing documents can give an impression or provide evidence. Interviewing employees allow the possibility to keep asking through when an employee is describing a CSF or practical example of the CSF, this might contribute to get a better understanding of the CSF.

Because of the limited time for this research it is not possible to carry out this research on different moments in time, that's why there is chosen for a cross-sectional time horizon. This means that the research will be conducted at one particular moment in time. (Saunders et al., 2016)

3.2 Technical design: elaboration of the method

A case study is an in-depth inquiry into a topic or phenomenon within its real-life setting (Yin, 2014) and could consist of a single case or multiple cases. (Saunders et al., 2016) The "case" in case study research may refer to a person (e.g. a manager), a group (e.g. a work team), an organisation (e.g. a business), an association (e.g. a joint venture), a change process (e.g. restructuring a company), an event (e.g. an annual general meeting) as well as many other types of case subject. (Saunders et al., 2016)

The main study (of which this study is a part) conducted a similar study, but in terms of CSF's which apply to "Continuous Integration", in which 117 potential CSF's were found.

The 92 potential CSF's found in this study and the 117 potential CSF's which apply to Continuous Integration together have been classified by the principal investigator in collaboration with the supervisor on the basis of similarity. This has resulted in a newly compiled list of only 19 potential CSF's (Appendix U). This list of 19 potential CSF's will be used in the rest of this study and apply to both Continuous Integration and Continuous Delivery / Continuous Deployment. Called: "Continuous Practices".

There is chosen to conduct a single case study, in order to verify the list with potential CSF's. The reason there is chosen to conduct a single case study is because the DevOps context (DevOps team) is necessary in order to explore how the CSF's are addressed in the organization and can be marked as "the case" in this research. The sample size will consist of one organization in order to explore as much information as possible. The data that is required to verify the list with potential CSF's is outlined in table 4.

Table 4 - Data required per potential CSF

Relates to subquestion	**Data required**
SQ3 - CSF criteria 1 - Recognition of the CSF	A practical example concerning the CSF
SQ3 - CSF criteria 2 - At least one verifiable performance improvement, incident or success	A practical example of the performance improvement caused by the CSF
SQ3 - CSF criteria 3 - Verification of the way of measurement of the CSF	Providing (measurement) report(s) or document(s) concerning the performance improvement

The unit of analysis in this research will be an organization which contains the following selection criteria:

1. Provides IT services to (end) users and focuses on software production
2. Has an organization size of > 1000 employees
3. Established (a) DevOps team(s) which consists of software development team(s) and operation team(s)
4. Has implemented Continuous Integration, Continuous Delivery and/or Continuous Deployment with a minimum of 2 - 3 years ago
5. Continuous Integration, Continuous Delivery and/or Continuous Deployment pipeline process steps, role distribution and responsibilities could be clearly explained
6. Has established shared goal(s) concerning Continuous Integration, Continuous Delivery and/or Continuous Deployment
7. Has established performance requirements which are applied to the established shared goal(s)
8. Minimum of 8 respondents to conduct interviews (by a margin/bandwidth of 2 respondents)

Criteria 1 is necessary to select organizations which provide IT services and focuses on software production, organizations who do not provide IT-services and do not focus on software production does not have to deal with DevOps or Continuous Integration, Continuous Delivery and/or Continuous Deployment.

Criteria 2 is established because the organization should not be too small, if it is assumed that in small organizations things are often done easier/faster because actions are often less dependent. Besides that, it is also expected that the process of Continuous Integration, Continuous Delivery and/or

Continuous Deployment is institutionalized better at bigger IT service providers, then small size IT service providers. Therefore certain CSF's may not be applicable.

Criteria 3 is established, because the organization has to be established DevOps teams according to the definition given by (Erich, 2019). This selection criteria is chosen because of the selected context of this research.

Criteria 4 is established because, if an organization is chosen with a shorter period of time between the implementation of Continuous Integration, Continuous Delivery and/or Continuous Deployment and this study, it will be very unreliable to determine the long term effect with regard to the performance improvements.

Criteria 5 is established because it is important that the researcher knows what the organization understands under the Continuous Delivery pipeline, role distribution and responsibilities, so that a good picture of the context can be created. If the organization is not able to indicate this clearly, this can be at the expense of the quality of this research.

Criteria 6 is established because it is important in this study that (a) shared goal(s) has/have been established to know for which shared goals Continuous Integration, Continuous Delivery and/or Continuous Deployment is used, which contributes to gather a good picture of the DevOps context.

Criteria 7 is established because it is important that performance requirements are established, which apply to these shared goal(s), to gather a more detailed picture of the DevOps context.

Criteria 8 is established because we want to collect as much information as possible and 8 respondents (with a margin/bandwidth of 2 respondents) seem sufficient to verify the entire list of potential CSF's.

The interviewees (respondents) must contain the following selection criteria:

1. Member of a DevOps team (taking into account the diversity of (T-)profiles in the DevOps team)
2. A minimum of 7 years of work experience in current role
3. A minimum of 5 years of work experience in current role inside the case organization
4. A minimum of 3 years of relevant work experience concerning Continuous Integration, Continuous Delivery and/or Continuous Deployment and DevOps
5. A minimum of 2 years of relevant work experience concerning Continuous Integration, Continuous Delivery and/or Continuous Deployment and DevOps inside the case organization
6. Has a positive attitude towards Continuous Integration, Continuous Delivery and/or Continuous Deployment and DevOps (believes that Continuous Integration, Continuous Delivery and/or Continuous Deployment and DevOps are successful implemented inside the case organization)
7. Permission to participate in the interview

Criteria 1 is established because the respondent has to be part of the DevOps team.

Criteria 2-5 are established because we want to interview experts which are fully accustomed to the organization and have enough experience, to maximize the quality of this study.

Criteria 6 is established because when most of the respondents (80%) have a negative attitude towards Continuous Integration, Continuous Delivery and/or Continuous Deployment and DevOps it may be possible that the adoption or implementation phase is not passed yet or incomplete, which may indicate that the organization is not a suitable case organization.

Criteria 7 is established because if none of the respondents grant permission to participate in the interviews, the organization is not a suitable case organization.

The data will be gathered by conducting semi-structured interviews. The semi-structured interviews will be conducted in Dutch when the interviewee is Dutch, otherwise interviews will be conducted in English as a second language. All interviews will be recorded in terms of audio-recordings. The method to verify the list of CSF's during the semi-structured interviews consists of three steps.

In the first step, a pre-interview will be taken from the gatekeeper of a candidate organization in order to check if the organization is a suitable candidate to participate in this research. This will be checked against the predefined selection criteria for an organization. In addition, information about the DevOps context will also be collected (e.g. expertise, role distribution, responsibilities, shared goals and performance requirements). During the interview with the gatekeeper, there will be asked which shared goals the DevOps team has set with regard to Continuous Integration, Continuous Delivery and/or Continuous Deployment and which performance requirements (with possible measurable units) apply to these set of shared goals. This information will be used in order to create a contextual picture and to know if we can speak of a successful implementation of Continuous Integration, Continuous Delivery and/or Continuous Deployment when the shared goal(s) and/or performance requirement(s) is/are achieved. After the pre-interview the gatekeeper will be asked if he could propose some interview-candidates who might want to participate in this research.

In the second step, a list of 7 questions will be mailed to the interview-candidates in order to test the interview-candidates against the selection criteria for a respondent. In the last question the interview-candidate will be asked if he/she wants to participate in this research. If there are enough respondents the research will continue at the selected organization.

In the third step the interviewer will conduct the interviews with the selected respondents, the respondents will be asked if they recognise the proposed CSF within their team. If they recognize the CSF, the respondent will be asked if he/she can describe/demonstrate at least one verifiable incident or success by means of a practical example, (measurement) reports or documents. After that, the respondent will be asked about the weight and his/her substantiation why he/she thinks the CSF is important or not. If the respondent is able to describe any performance improvements, the interviewer will ask in what way they measured the performance improvement(s). If there is enough time available, the respondent will be asked which action / approach was taken concerning the CSF.

The full interview guide is available in the interview-protocol, which can be seen in Appendix S. As soon as all interviews are completed, the audio recordings will be transcribed so the data could be used for analysis. Transcription of the semi-structured interviews will be done in the language which the interview was conducted in order to maximize the feedback value of the interviewee when validating the interview. Coding will be in Dutch and further analysis of the interviews will be in English.

3.3 Data analysis

The data concerning the organization selection criteria will be coded (see table 5) and will be sorted by selection criteria. For each selection criteria the data (quotations) will be synthesized, and a brief conclusion will be given per selection criteria. Lastly, a final conclusion will be given, why the organization is selected as suitable case organization.

The data concerning the respondent selection criteria will be collected by email and will be sorted without coding, because the amount of data is very small and can be seen at a glance. Lastly, a final conclusion will be given, which respondents are selected to participate in the interviews.

The data concerning the potential CSF's will be analyzed by making use of a "deductive approach" by comparing the empirical observations with the list of potential CSF's which were found during the theoretical research (theoretical expectations) in order to verify the list of potential CSF's.

The verification of the potential CSF's will be done in order of strength. First of all, each potential CSF will be analyzed by means of their definition, in order to know the potential CSF is well understood. Secondly, each potential CSF will be analyzed by means of their presence, in order to know the potential CSF has been recognized/experienced. Thirdly, each potential CSF will be analyzed by means of the presence of a practical example concerning the potential CSF, in order to know the potential CSF has met criteria 1 of a CSF. Fourthly, each potential CSF will be analyzed by means of their weight indication, in order to know how important the potential CSF might be. Fifthly, each potential CSF will be analyzed by means of their substantiation of the weight indication, in order to analyze the presence of a performance improvement concerning the potential CSF and to know the potential CSF has met criteria 2 of a CSF. Sixthly, each potential CSF will be analyzed by means of the presence of a measurement report or document concerning the way of measurement of the potential CSF, in order to know the potential CSF has met criteria 3 of a CSF. Seventhly, each potential CSF will be analyzed by means of their implementation, in order to know how the potential CSF has been addressed. Lastly, each potential CSF will be analyzed by means of the presence of a contextual relationship with a shared goal or performance requirement, in order to know which potential CSF's contribute to a shared goal or performance requirement.

The data analysis of the transcriptions will be done by the use of coding. For coding the transcriptions, Atlas.TI will be used as coding software.

According to Strauss and Corbin there are three coding stages: the reorganisation of data into categories is called open coding, the process of recognising relationships between categories is referred to as axial coding and the integration of categories to produce a theory is labelled selective coding. (1998) Open coding (Strauss & Corbin, 1998) is used to highlight the evidence (e.g. criteria 1, criteria 2 and criteria 3 concerning the CSF's in the DevOps context) to provide an answer to the main research question.

The coding of these transcriptions will be done in Dutch and consist of a category (particular CSF), a code and a sub-code which will be merged to one code, the coding scheme is shown in table 5 in Appendix AA and is translated into English. The column "Relates to question" refers to the number of the question during the interview, according to the interview-protocol (Appendix S). The column "Code" refers to the name of the code. The column "Sub-code" refers to the name of the sub-code. The *{name}* in the sub-code attribute refers to a concise interpretation of the descriptions in the definition attribute. The column "Definition" refers to a description of the highlighted evidence in the transcription.

After the coding is complete, the data will be analysed by synthesising the data (quotations) with the same code, because it is possible that there are multiple quotations refer to one unique code. This can be done by Atlas.TI by making a spreadsheet export file, so it is easy to filter on a certain code, to have the results all together (synthesized). Data concerning the organization selection criteria will be synthesized in the structure according to table 6 in Appendix AA.

Data concerning the respondent selection criteria will be synthesized in a single table. In order to provide results which will be used in order to answer the main research question, the data concerning the potential CSF's will be synthesized per CSF in the structure according to table 7 in Appendix AA.

After synthesizing the data, a potential CSF has to provide evidence in the form of quotations given by (a) (the) respondent(s) about the following:

- Criteria1 (Provide a quotation of an example of the CSF in practice)
- Criteria2 (Provide a quotation, which indicates the performance improvement)

- Criteria3 (Provide a quotation of the way of measurement by means of a measurement report or document)

When all three criteria of a potential CSF are verified, the potential CSF could be considered as a verified Critical Success Factor.

Contextual relationships that could arise between certain codes could be CSF's which relate to performance requirements or shared goals. In addition, relationships could also arise between Continuous practices which contain mutual interdependence with DevOps.

3.4 Reflection w.r.t. validity, reliability and ethical aspects

The research design is set up in a sound and prudent manner by taking into account the construct validity (which means according to Yin (2008) "establishing correct operational measures for the concepts being studied."), external validity (which means according to Yin (2008) "*establishing the domain to which a study's findings can be generalized.*") and reliability (which means according to Yin (2008) "*demonstrating that the operations of a study —such as the data collection procedures can be repeated, with the same results.*").

The internal validity is not taken into account because according to Yin (2008) internal validity is not applicable to exploratory studies: "*for explanatory or causal studies only, and not for descriptive or exploratory studies): establishing a causal relationship, whereby certain conditions are shown to lead to other conditions, as distinguished from spurious relationships.*" (Yin, 2008)

Because the CSF's have been operationalized by the author only, good construct validity can be confirmed because test-interviews were taken into account. Test-interviews should ensure that operationalization errors are identified. (e.g. interpretation difference with the author of the study from which the CSF was taken) In addition, the list of potential CSF's are reviewed by an experienced professional in the field of DevOps / Continuous Delivery to maximize construct-validity. In addition, for each CSF the definition will be discussed and checked with the respondent if he/she has the same interpretation of the definition as the interviewer.

Good reliability can be confirmed by the fact that an in-depth methodological description is presented by having an interview-protocol to allow the study to be repeated. This interview-protocol can be seen in Appendix S.

Despite the fact that it was decided to interview one case organization, the external validity is therefore a point of weakness. In the outcome of this study it should be taken into account that the external validity of this case study can be generalised in a smaller degree.

Ethics refers to the standards of behavior that guide a researcher's conduct in relation to the rights of those who become the subject of work, or affected by it. (Saunders et al., 2016). This research takes into account the following ethical principles as listed by Saunders et al. (2016) these are: "integrity and objectivity of the researcher", "respect for others", "avoidance of harm (non-maleficence)", "privacy of those taking part", "voluntary nature of participation and right to withdraw", "informed consent of those taking part", "ensuring confidentiality of data and maintenance of anonymity of those taking part", "responsibility in the analysis of data and reporting of findings", "compliance in the management of data" and "ensuring the safety of the researcher" by having the respondents sign a letter of consent before the start of the interview. (appendix W)

In addition, I am also committed to the Netherlands Code of Conduct for Research Integrity, 2018 (appendix Q) and have signed "Declaration Own Work October 2017" (Appendix R).

4. Results

This chapter describes the implementation of the research. First of all the unit of analysis will be described, which contains the data gathered during the execution of the gatekeeper interview, the execution of the selection of the respondents and the interviews with the selected respondents, all to check if they met the corresponding selection criteria. Secondly, the execution of the pilot interview and the execution of the interviews with the selected respondents will be described. Finally, the results concerning the list of potential CSF's will be described.

4.1 Unit of analysis - Selection of case organization

To assess whether the case organization is suitable, a gatekeeper interview was conducted with the operational manager and a teamleader of the case organization. The raw data concerning the organization selection criteria can be seen in Appendix Z. A detailed description of the answers concerning the selection criteria and the context can be seen in Appendix ZB.

The case organization has several government parties that are clients for the delivery of custom applications. The implementation of the legislation is generally realised by means of customised packages (self-built software). The case organization has about 3000 employees, of which between 80-85 % are internal- and 15-20 % external employees. Based on this information we can conclude that the case organization meets the first and second selection criteria.

The definition given by the gatekeeper is also inline with the definition of DevOps given by Erich (2019) we adopt in this study: "interaction between development and operations". Inside the case organization there are between 45 and 50 DevOps teams assigned. Based on this information we can conclude that the case organization also meets the third selection criteria. The case organization has established (a) DevOps team(s) which consists of software development team(s) and operation team(s).

The gatekeeper indicates that Continuous Delivery is on the same level as Continuous Integration, Continuous Integration was commonplace 5/6/7 years ago and Continuous Delivery for them over the past 2-3 years has actually reached the level that they can say they are ready in every team. Based on this information we can conclude that the case organization also meets the fourth selection criteria. The case organization has implemented Continuous Integration, Continuous Delivery and/or Continuous Deployment with a minimum of 2 - 3 years ago.

According to Appendix ZB we can conclude that the case organization also meets the fifth selection criteria. The Continuous Integration, Continuous Delivery and/or Continuous Deployment pipeline process steps, role distribution and responsibilities could be clearly explained.

According to Appendix ZB we can conclude that the case organization also meets the sixth selection criteria. The case organization has established shared goals concerning Continuous Integration, Continuous Delivery and/or Continuous Deployment.

According to Appendix ZB we can conclude that the case organization also meets the seventh selection criteria. The case organization has established performance requirements.

The gatekeeper has proposed 6 interview candidates. Based on this information we can conclude that the case organization also meets the eighth selection criteria. The case organization proposed a minimum of 8 respondents to conduct interviews (by a margin/bandwidth of 2 respondents).

Because the case organization meets all selection criteria, this case organization was selected to conduct this research. The gatekeeper was also requested to provide the document that shows the target situation of the context of the case organization with regard to DevOps, Continuous Integration, Continuous Delivery and/or Continuous Deployment. This can be seen in Figure 6 in Appendix ZB.

4.2 Unit of analysis - Selection of the respondents

The gatekeeper proposed candidate respondents to conduct interviews with. These candidate respondents had received an email with a number of questions according to the interview protocol (Appendix S) to check if they met the respondent selection criteria. See Appendix Y for the raw data concerning the respondent selection criteria, table 8 in Appendix AA shows a simplified view.

According to table 8 in Appendix AA, the respondents have met selection criteria 1, because all respondents are members of a DevOps team. Except for respondent 2, who used to be in a service team, (The Continuous-delivery team) and would like to speak to me from that role, because he has been one of the key figures, regarding the choice for Continuous practices.

According to table 8 in Appendix AA, the respondents have met selection criteria 2 also, all respondents have a minimum of 7 years of work experience in their current role. At selection criteria 3, there is one respondent which has 4 years of experience instead of a minimum of 5 years of work experience in their current role inside the case organization. This will be accepted because of the limited respondents which wanted to participate. Besides, 1 year less experience isn't a big deal. On the other hand, he/she has 7 years of experience with DevOps and Continuous practices.

According to table 8 in Appendix AA, all respondents have a minimum of 3 years of relevant work experience concerning Continuous Delivery and/or Continuous Deployment and DevOps and a minimum of 2 years of relevant work experience concerning Continuous Delivery and/or Continuous Deployment and DevOps inside the case organization. So selection criteria 4-5 has also been met.

According to table 8 in Appendix AA, most respondents had a positive attitude towards Continuous Delivery and/or Continuous Deployment and DevOps and believes that Continuous Delivery and/or Continuous Deployment and DevOps are successfully implemented inside the case organization, except for two respondents which are likely in doubt. One respondent is saying: "*We're definitely on the right track. In my opinion, we are not yet far enough advanced with test automation for continuous delivery. At least with regard to our project.*" Another respondent is saying: "*I don't know if this is the Holy Grail or not.*" Because no respondents had a negative attitude towards Continuous Delivery and/or Continuous Deployment and DevOps selection criteria 6 has also been met.

All respondents already had granted permission to the gatekeeper by saying that they wanted to participate. So selection criteria 7 has also been met, which implies that the case organization is suitable to proceed with the research.

4.3 Execution of the (pilot) interview(s)

Appointments were made with the selected respondents to conduct the interviews. First of all, a pilot interview was carried out to see whether the interview protocol (Appendix S) still needed to be adjusted. This was not necessary. The pilot interview did, however, provide hints about the way of interviewing.

For example, the respondent was given a preference for the starting point on the list of potential CSF's. Some respondents had already printed out the list of potential CSF's and made notes. This made it

immediately clear from which CSF's they knew a lot about and which they never had experienced or were unclear. You could deduce from this that most respondents had been well prepared.

Verifying the definitions of the potential CSF's at the start of the CSF was also more convenient because it was easier to do this immediately instead of finding out during the CSF that the definition was not clear enough.

After the pilot interview, the other interviews were conducted. Before the start of the interview, the consent form was signed and permission was requested from the respondent to allow the interview be recorded by means of an audio recording. The duration of the audio recordings was between 74 - 103 minutes. Before the audio recording was started, the researcher first introduced himself by means of a short introduction about the educational programme and purpose of his research. This usually took no more than 15 minutes. Reservations were often made for 1.5 hours. Some respondents indicated that they had even more time and wanted to complete the full potential CSF list. A transcript was made of each interview and sent to the respondent for verification. All respondents have verified that these have been correctly transcribed.

4.4 Results

The raw data concerning the data synthesis of the potential CSF's, can be seen in Appendix Z.

In order to strengthen the results, only examples were selected of respondents who experienced the potential CSF, were able to give a reasonable example of the CSF in practice and were also able to give a reasonable substantiation of the weight indication. All results concerning the verification of the list of potential CSF's including a detailed description and supporting quotations can be seen in Appendix ZA.

CSF - Preconditions

Example: Respondent 3 indicates as an example of a precondition that he is often dependent on other teams.

Substantiation: The reason why respondent 3 thinks this CSF scores a 5 and is therefore seen as very important is because he/she indicates that it is his/her concern, towards the user. He/she considers him/herself responsible for the result, he/she has sent to that user. Therefore he/she thinks it is inappropriate that the user has to wait very long.

Example: Respondent 4 indicates as an example of a precondition that the "use case descriptions" and "UX-designs" need to be defined, because they are not always clear and often change.

Substantiation: The reason why respondent 4 thinks this CSF scores a 3 and is therefore seen as relatively important is because he/she indicates that by defining the design for a sprint, they probably can achieve the sprint.

CSF - Goals

Example: Respondent 2 indicates that they were guided by the company Xebia Labs at the beginning of their Continuous Delivery journey using the deploy tool: "XL Deploy" which contains "Continuous Delivery Maturity Metrics", according to the respondent, goals could result from this if it appears that certain aspects of Continuous Delivery score low. After that, the respondent indicates that they have used the "Continuous Delivery Maturity Metrics" to clarify the management where to put effort and resources.

Substantiation: The reason why respondent 2 thinks this CSF scores a 3 and is therefore seen as relatively important is because he/she indicates that it results in focus. He/she indicates that it could

be used as an instrument to convince the management of where the focus should be at the moment.

Measurement: Respondent 2 indicates that the "Continuous Delivery Maturity Matrix" is just a .pdf document, where you have to draw a line when the description of the maturity level is equal to the reality.

CSF - Strategy and approach

Example: Respondent 2 gives as an example that the case organization used a tech-driven approach, which can be seen as a bottom-up approach.

Substantiation: The reason why respondent 2 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that a strategy is not necessary and a bottom-up approach is important because Continuous Integration and Continuous Delivery is a relatively technical story and often does not receive the necessary attention from management and is therefore not imposed from above in the form of a strategy.

Measurement: Respondent 2 indicates that technical architecture documents are being used for the approach.

CSF - Architecture

Example: Respondent 1 indicates that they have a separate club of architects, who create the architecture. Six months ago they found out that the architecture is very well designed, but the infrastructure is not yet there at all. So their plans for what they wanted are there, but it has not yet been realized, so there is a discrepancy between the designs and reality. This example is also confirmed by respondents 3 and 5.

Substantiation: The reason why respondent 5 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that if you stay within the established boundaries of the architecture, you can simply deliver faster. Because you don't have to discuss everywhere. According to the respondent, it takes a lot of time and lobbying to get things done that fall outside the boundaries. In that sense he thinks architecture is important.

Substantiation: The reason why respondent 1 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that it contributes to develop things a lot faster.

Performance improvement: Respondents 1 and 5 indicate in their substantiation that this potential CSF will imply speed.

CSF - Process design

Example: Respondent 6 indicates that the process design is embedded in the tooling.

Substantiation: The reason why respondent 6 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that having such a process can be seen as a prerequisite.

Example: Respondent 5 indicates that they have a full support team working on the tooling around enabling Continuous Integration and Continuous Delivery "The Continuous Delivery team" and, according to the respondent, is institutionalized during departmental consultation meetings.

Substantiation: The reason why respondent 5 thinks this CSF scores a 5 and is therefore seen as very important is because he/she indicates that it is very important to have a process, because he/she does not have the expertise to do it on their own.

Example: Respondent 3 indicates that there is a provisioning thing with which you can create a silo and where components can be added to it. From that silo you can, according to the respondent, perform tests, check in code in a repository, build, indicate which checks need to be done in the Jenkins pipeline and automatically transfer it to a production silo via certain tooling.

Substantiation: The reason why respondent 3 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that the process design makes sure that you don't step into certain pitfalls, which you can easily skip without that tooling. Things can slip through. For example security aspects. The process design that forces you to follow the process.

Example: Respondent 2 indicates that they are developing a “pipeline template” by means of the “Jenkins template engine” which ensures that hack tests will be institutionalized.

Substantiation: The reason why respondent 2 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that the freedom you have, if you don't institutionalize it, leads to abuse.

CSF - Motivation

Example: Respondent 4 indicates that his team monitors the build status properly.

Substantiation: The reason why respondent 4 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that a lack of motivation could possibly negatively affect other team members.

Example: Respondent 3 indicates that they strive to check-in their work at least once a day, regularly runs tests and if you don't know something, asks someone else for help.

Substantiation: The reason why respondent 3 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that the team effort is important so that the knowledge is also distributed across the team.

Example: Respondent 2 indicates that people still fix issues on an individual basis.

Substantiation: The reason why respondent 2 thinks this CSF scores a 5 and is therefore seen as very important is because he/she also indicates that team effort, team responsibility and ownership contribute to the right DevOps culture.

CSF - Resistance to change

Example: Respondent 6 indicates that there is some kind of mismatch between the responsibility and authorization of team members of the DevOps team.

Substantiation: The reason why respondent 6 thinks this CSF scores a 3 and is therefore seen as relatively important is because he/she indicates that their current way of working is not impossible either, it's just annoying, they have to ask other people to do things for them every time. But it's not pure need, they can live with it.

Example: Respondent 4 indicates that security aspects such as hack testing do not cooperate conveniently with regard to Continuous Delivery because they still have to request hack tests and an inquiry usually takes about 6 - 8 weeks.

Substantiation / Performance improvement: The reason why respondent 4 thinks this CSF scores a 5 and is therefore seen as very important is because he/she indicates that you won't achieve the maximum result of Continuous Integration / Continuous Delivery if the DevOps team is not able to perform hack tests by themselves. He/she also indicates that performing hack tests themselves within the DevOps team would result in more speed, in order to go faster through the street.

Example: Respondent 2 indicates that there is a mismatch in the current organizational policy which causes a lack of application ownership.

Substantiation: The reason why respondent 2 thinks this CSF scores a 5 and is therefore seen as very important is because he/she indicates that by changing the organizational boundaries, "the technical application administrators" cannot be considered as a separate division anymore, but is now part of the DevOps team.

CSF - Complexity across customer organization boundary

Example: Respondents 6 and 4 indicate that they have no access to production, which means they don't have the ability to analyze what's going on when something malfunctions.

Substantiation: The reason why respondent 6 thinks this CSF scores a 1 and is therefore seen as unimportant is because he/she indicates that the tooling does have access to production and other people within his team also have access. So it is unimportant to him/her to have access by themselves.

Substantiation: The reason why respondent 4 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that if they can do more on their own, they might be able to fix it faster. Now they depend on third parties to look into production.

Example: Remarkably, respondent 2 indicates that he/she does not consider it necessary to have access to the production environment, because the environments are identical and they have tooling which can make extractions of the data so that they can simulate the production environment.

Substantiation: The reason why respondent 2 thinks this CSF scores a 1 and is therefore seen as unimportant is because he/she indicates that they do not consider production as the end point and have all the conditions in place in such a way that they are also independent of it.

CSF - Acceptance by customer

Example: Respondent 6 indicates that they go to production every 3 weeks and the customer just accepts that. This happens outside office hours, in the morning between 6 and 7. According to the respondent, this is just a small moment. He/she also indicates that things will be different if, for example, the customer is out every afternoon between 2 and 3 because of a small feature that the customer doesn't even experience.

Substantiation: The reason why respondent 6 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that it's important for the reputation of the organization.

Example: Respondent 3 indicates that in his/her current project they introduced Slack groups, which are a kind of message medium, in which they are in very brief contact with the customers of their services. Wishes and problems are then placed in such a Slack group and they can all respond to them. And make sure that the problem is solved. He/she indicates that this increases customer involvement and that the customers also accept this way of working.

Substantiation: The reason why respondent 3 thinks this CSF scores a 5 and is therefore seen as very important is because you make something for the customer, you want to have answered the

customer's request properly and you can only do that if that line is short, that you also understand what the customer is struggling with and exactly know what he wants.

CSF - Sales and intermediaries

Example: Respondent 4 indicates that user data is not accessible from colleagues in The Hague.
Substantiation: The reason why respondent 4 thinks this CSF scores a 5 and is therefore seen as very important is because it takes them a lot of extra time and money to get the same result.

CSF - Quality

Example: Respondent 4 indicates that the quality of the code with comments, is much more important than the functional documentation. This is also confirmed by respondent 3, he/she also indicates that he/she doesn't accept the review when there is no adequate documentation inside the code.

Substantiation / Performance improvement: The reason why respondent 3 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that adequate documentation provides speed, because other people don't have to dive into the code.

Example / Measurement: Respondent 6 indicates that they measure the degree of test coverage of the code to preserve quality coding. In addition respondent 6 also indicates that there are all kinds of tools which can help you to ensure the quality of your code.

Substantiation: The reason why respondent 6 thinks this CSF scores a 3 and is therefore seen as relatively important is because he/she indicates that quality is not always crucial, timely delivery for example, can be even more important.

CSF - Customer involvement

Example: Respondent 5 indicates that they use a "Sprint 0", in which the customer is present, so that the customer's wish can be clearly determined.

Substantiation: The reason why respondent 5 thinks this CSF scores a 5 and is therefore seen as very important is because he/she indicates that that you're probably making something that wasn't intended and therefore doesn't meet the quality standards, which could result in dissatisfied customers.

Example: Respondent 2 indicates that Business Analysis Teams (BAT) are receiving customer input and feedback and provide this information to the DevOps teams. This is also their responsibility. They also create the user stories and are involved from the beginning of the process. This example is also confirmed by respondent 4.

Substantiation: The reason why respondent 2 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that for example "the ability to deliver faster", improved the culture in the sense that the customer accepts that effort is needed that has to be invested in the technical side of Continuous Delivery, instead of telling the customer a technical story.

Substantiation / Performance improvement: The reason why respondent 4 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that customer involvement implies speed.

Example: Respondent 3 indicates that an information model has been developed as a result of frequent consultation with stakeholders.

Substantiation: The reason why respondent 3 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that it is important that the customer and the producer can work from the same image.

CSF - Test complexity & source code control

Example: Respondent 6 indicates that test automation is very important, because as soon as a new part of functionality has been created, you as a tester will never manually go through everything again. He/she also indicates that Cucumber is used to test applications and test automation could be seen as a prerequisite for Continuous Integration and Continuous Delivery.

Substantiation: The reason why respondent 6 thinks this CSF scores a 3.5 and is therefore seen between relatively important and important is because he/she considers the building of automatic test environments to be slightly important, because it might take a little more time, but they'll make it somehow. On the other hand he/she indicates the automation of tests to be very important, because he/she indicates that automatic testing is a prerequisite for Continuous Delivery.

Example: Respondent 4 indicates that they write unit-tests on their own. He/she also indicates that there is a seperate team called “test automation” which has made it relatively easy for them to build their own integration-tests and regression-tests.

Substantiation / Performance improvement: The reason why respondent 4 thinks this CSF scores a 5 and is therefore seen as very important is because he/she indicates that because of the complex code these days, that it can't really be tested anymore without automatic testing. By use of automatic testing, testing becomes easier and faster.

Example: Respondent 3 indicates that the complexity lies in the management of test environments, which is caused by the amount of test environments.

Substantiation: The reason why respondent 3 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that without source code control it is inconvenient and not useful for cooperation.

CSF - Coordination

Example: Respondent 6 indicates that there is a need for coordination to align the version numbers of different functionalities used by different applications.

Substantiation: The reason why respondent 6 thinks this CSF scores a 5 and is therefore seen as very important is because coordination is necessary to align the different applications, so they can communicate between them.

Example: Respondent 4 indicates that there is a need for coordination to align the technology behind the applications between the different teams.

Substantiation: The reason why respondent 4 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that coordination is necessary to establish one uniform standard for software, which is a lot easier.

CSF - Communication

Example: Respondent 3 indicates that “Skype” is used as a communication tool to facilitate intra and inter team communication.

Substantiation: The reason why respondent 3 thinks this CSF scores a 3 and is therefore seen as relatively important is because he/she indicates that transparency can serve to improve the team.

Example: Respondent 6 indicates that everyone in his/her team physically sits together and communicates very openly to each other and during the daily stand-up, they discuss what everyone is working on. He/she also indicates that JIRA is used to create stories and to watch/trace issues.

Substantiation: The reason why respondent 6 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that it is important to know from each other what everybody is doing.

Example: Respondent 4 indicates that the scrumboard is also often present on paper, so they can see what each team is doing and what its current status is.

Substantiation: The reason why respondent 4 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that without communication, you don't know when something is finished which is built by another team, where you are waiting for.

CSF - Knowledge and training

Example: Respondent 6 indicates that knowledge is definitely available within his/her team.

Substantiation: The reason why respondent 6 thinks this CSF scores a 3 and is therefore seen as relatively important is because he/she indicates that not everyone in the team should know exactly how everything works, but he/she indicates that they are making use of a service team which carry out the deployment of application servers which are being used by the team. However, by working together with a service team, knowledge is kept away.

Example: Respondent 4 indicates that he/she is busy with a POC and following workshops to experiment with certain web technologies in order to decide if this is interesting for the case organization.

Example: Respondent 3 also indicates that information transfer also happens at the desk and sometimes there are pizza sessions.

Substantiation: The reason why respondent 3 and respondent 4 thinks this CSF scores a 4 and is therefore seen as important is because they indicate that when you don't maintain your area of expertise, then at some point you will not be part of the game anymore.

CSF - Tooling

Example: Respondent 6 indicates that there is a list of tools which are being used to support the Continuous Delivery process. "IDE", "Intelie", "Jenkins", "XL-Deploy", "XL-Release", "OWASP" and "Fortify". He/she also indicates that some tools are very strict, when it comes to scanning of the code, to check if there are bugs in it, like Fortify, which causes a lot of false positives and therefore takes a lot of time.

Substantiation: The reason why respondent 6 and respondent 5 thinks this CSF scores a 5 and is therefore seen as very important, is because they indicate that tooling can be seen as a prerequisite for Continuous Integration, Delivery and/or Continuous Deployment.
Example: Respondent 3 indicates that he/she sometimes has questions about the maturity of the tools they use, according to him/her the interfacing of "XL Deploy" is not that handy.

Substantiation: The reason why respondent 3 thinks this CSF scores a 5 and is therefore seen as very important is because he/she indicates that tooling is also a prerequisite especially to get enough support to implement this in the organization.

CSF - Pace

Example / Performance improvement: Respondents 4 and 5 indicate that the speed of delivering deployments to the customer is increased by releasing (incremental) new functionality every 3 weeks, by comparing this example with a method from the past.

Substantiation: The reason why respondent 4 thinks this CSF scores a 2 and is therefore seen as slightly important is because he/she indicates that the quality of the software takes precedence over speed.

Substantiation: The reason why respondent 5 thinks this CSF scores a 5 and is therefore seen as very important is because he/she indicates that in comparison with a method from the past, the increased delivery speed may have influence on the customer satisfaction.

CSF - Pressure

Example: Respondent 6 indicates that he/she has experienced pressure because of deadlines, which forced them to make concessions to quality, which was the expense of code reviews and duplication of code at one place.

Substantiation: The reason why respondent 6 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that if the team doesn't experience pressure, they can deal with it in a relaxed way, which increases the risk of not being able to deliver on time.

Example: Respondent 4 indicates that it depends on how you deal with situations when the customer asks something from you. If you experience that as pressure or not.

Substantiation: The reason why respondent 4 thinks this CSF scores a 3 and is therefore seen as relatively important is because he/she indicates that he/she does not let himself/herself be under pressure so quickly.

Example: Respondent 1 indicates that pressure is caused by deadlines, that pressure is caused by interference of the customer and that the pressure they are experiencing is caused by factors beyond their control, like other teams they depend on.

Substantiation: The reason why respondent 1 thinks this CSF scores a 3 and is therefore seen as relatively important is because he/she indicates that pressure makes sure there's no loafing.

Measurement: Respondent 1 indicates that maybe pressure could be measured in story points, but it remains a bit of a guess.

Weight indications of the potential CSF's

According to table 10 it seems that the weight indication given by the respondents for all the potential CSF's seems to be very diverse. The column "Potential CSF Name" refers to the name of the potential CSF according to Appendix U. The columns "R{x}" refers to the weight indication on a scale of 1-5 provided by the respondents (alias). An empty value means that the respondent did not give a weight indication of the potential CSF.

Table 10 - Weight indication of the potential CSF's given by the respondents

Potential CSF Name	R1	R2	R3	R4	R5	R6

Preconditions			5	3		5
Goals		3		1	4.5	4
Strategy and approach		4		4	1	
Architecture	4	3	4	4	4	2
Process design		4	4	4	5	4
Motivation		5	4	4	5	4
Resistance to change		5	3	5	4	3
Complexity across customer organization boundary		1		4		1
Acceptance by customer			5		5	4
Sales and intermediaries				5		
Quality			4	4	5	3
Customer involvement	4	4	4	4	5	4
Test complexity & source code control	4	2	4	5	4	3.5
Coordination	2	4	3	4	3	5
Communication	5	4	3	4	5	4
Knowledge and training			4	4		3
Tooling	4		5	5	5	5
Pace			2	2	5	
Pressure	3		4	3	2	4

Definitions of the potential CSF's

According to Appendix ZA, in general all definitions of the potential CSF's seems to be well understood. During the analysis it was observed that the definition of the potential CSF - "Pace", was not quite suited, because respondent 6 indicates that he/she did not experience this CSF is because he/she usually release every 3 weeks in a continuous flow, so it is impossible to experience an improved speed of delivery. Besides that, he/she indicates that it is also possible that there is more functionality in a single release, which can also be considered as an improved speed of delivering deployments to the customer. Based on this observation it can be deduced that the definition of the potential CSF - "Pace" needs to be tightened.

Experience of the potential CSF's

According to Appendix ZA, all potential CSF's have been experienced within the case organization.

Observations

During the analysis it was also observed that some respondents mentioned contextual relationships with CSF's. The following potential CSF's: *"Preconditions"*, *"Motivation"* and *"Pace"* seem to contain contextual relationships between the CSF and the "velocity" of DevOps teams. Descriptions of the contextual relationships with the CSF's can be seen in Appendix ZA. During the analysis it was also observed that the potential CSF - "Test complexity & source code control" might have a mutual interdependence between Continuous Integration and DevOps. One of the respondents indicates that: "*If you don't put your code in a central repository, it could mean people can't collaborate properly.*"

Final conclusion

Referring to sub question 3 which remained unanswered, in order to know how these potential CSF's have been addressed in a DevOps context and subsequently led to a successful adoption and implementation of Continuous Delivery and Continuous Deployment. The information about the criteria in order to verify the potential CSF's, which has been retrieved during the follow-up research can be seen in table 11. The column "Potential CSF Name" refers to the name of the potential CSF according to Appendix U. The column "Criteria 1" refers to the presence of an example of the CSF in practice. The column "Criteria 2" refers to the presence of a performance improvement of the CSF. The column "Criteria 3" refers to the presence of an example of the way of measurement of the CSF. In the case of a green cell, the criterion has been verified. In the case of a red cell, the criterion could not be verified.

Table 11 - Potential CSF's addressed in a DevOps context

Potential CSF Name	**Criteria 1**	**Criteria 2**	**Criteria 3**
Preconditions			
Goals			
Strategy and approach			
Architecture			
Process design			
Motivation			
Resistance to change			
Complexity across customer organization boundary			
Acceptance by customer			
Sales and intermediaries			
Quality			
Customer involvement			
Test complexity & source code control			
Coordination			
Communication			
Knowledge and training			
Tooling			
Pace			
Pressure			

5. Discussion, conclusions and recommendations

5.1. Discussion - reflection

According to Saunders et al. (2016) participation bias refers to the amount of time required for an interview may result in a reduction in willingness to take part by some. In general the respondents enjoyed participating in the interviews. They had prepared themselves very well. Some respondents had even printed out the list of potential CSF's and wrote down any ambiguities or circled potential CSF's they recognized. The interviews were scheduled for 1.5 hours, some respondents even indicated that they have more time and even wanted to go through the full list of potential CSF's in order to assist this research. From this it can be deduced that there was as little participation bias in this research as possible.

According to Saunders et al. (2016) interviewer bias is possible in the way you interpret responses. The interviewer often discussed the definitions of the potential CSF's with the respondent in advance of being questioned, in order to eliminate any differences in interpretation. The potential CSF's were well understood by the respondents. In addition, the interviewer also discussed the order in which the questions were asked and the purpose of this study.

From a target population point of view, according to the gatekeeper an average DevOps team has the following 4 traditional roles: "Designers", "Programmers", "Testers" and "Administrators". In this study only "Programmers" were selected to participate during the interviews. Because the gatekeeper could only bring in "Programmers" who were willing to participate in this research. This could also result in interviewer bias, because for example: "Testers" may think differently about certain potential CSF's, because their work may reflect other aspects of the Continuous Integration, Continuous Delivery and Continuous Deployment process than "Programmers".

However, the gatekeeper indicates that the "Source-control" phase of the delivery pipeline is primarily conducted by developers (programmers) and testers. In addition, the gatekeeper indicates that in practice the functions are somewhat more diffuse, because of the T-shape. The gatekeeper also indicates that you could say that those "package", "deploy" and "release" phases are work for the "administrators" in the team, but in practice, you see that developers play a role in these phases. The gatekeeper also indicates that the role as test manager or test coordinator can be filled quite well by a developer. Because of the T-shaped roles it can be derived that "developers" are playing a part in all phases of the Continuous Integration, Continuous Delivery and Continuous Deployment process. This indicates that, because of the T-shaped roles the selection of the sample to be only "Programmers" should be suitable.

By taking these measures, the interviewer has tried to counteract as much as possible interviewer bias in this research.

According to Saunders et al. (2016) interviewee or response bias can be introduced if interviewees may choose not to reveal and discuss an aspect of a topic that you wish to explore, because this would lead to probing questions that would intrude on sensitive information that they do not wish, or are not empowered, to discuss with you. The outcome of this may be that the interviewee provides only a partial 'picture' of the situation that casts himself or herself in a 'socially desirable' role, or the organization for which they work in a positive or even negative fashion. The interviewer has given the respondent a preference where to start on the list of potential CSF's or which potential CSF's he/she does/does not want to discuss. By taking these measures, the interviewer has tried to counteract as much as possible interviewee or response bias in this research.

Due to a lack of time, at the most potential CSF's, the interviewer was not able to ask about the way of measurement or implementation of most of the potential CSF's. This consideration was made because it seemed more important to reflect longer on examples and substantiations of weight indications of potential CSF's in order to maximise the quality of the responses so that the whole list of CSF's would have a stable foundation.

During the analysis it was observed that the definition of the potential CSF - "Pace" needs to be tightened. In this study we believe that the verification of criteria 1 and 2 of this potential CSF is still valid by carefully changing the definition of this CSF. Most of the respondents indicate that the speed of delivering deployments to the customer is increased by releasing (incremental) new functionality every 3 weeks, by comparing this example with a method from the past. Besides that, a respondent indicates that it is also possible that there is more functionality in a single release, which can also be considered as an improved speed of delivering deployments to the customer.

The new definition for this potential CSF may become: "*Improved speed of delivering deployments to the customer or delivering more functionality in a single sprint*"

5.2. Conclusions

The main research question of this study is: "Which Critical Success Factors apply to the adoption and implementation of Continuous Delivery and Continuous Deployment in a DevOps context?" It can be concluded that the full list of potential Critical Success Factors (CSF's) (Appendix U) has been recognized by the respondents and is supported with examples of practice. These examples imply that the first criteria in order to be a verified CSF, according to Leidecker & Bruno (1984) that a CSF should have a verified significant impact on the success of Continuous Delivery or Continuous Deployment, which means the factor is empirical verified has been met.

The following potential CSF's: *"Architecture", "Resistance to change", "Quality", "Customer involvement", "Test complexity & source code control"* and *"Pace"* results in verified performance improvements. These performance improvements imply that also the second criteria in order to be a verified CSF, according to Ram et al., (2013) that a CSF should result in verified performance improvements, has been met. All the observed performance improvements except *"Pace"* imply speed, which can be seen in more story points per sprint, because the respondents indicate that they have a continuous release schedule / flow of every 3 weeks. So if respondents indicate that: "the work is going to be faster", they imply that they could achieve more story points in a single sprint. This is also confirmed by a respondent: *"Because we just release once every 3 weeks. It is possible that in a single release, there is more functionality. So the release doesn't go any faster, but it could be that there is more in it."*

The observed performance improvement: *"Pace"* implies a continuous flow with shorter cycles as compared to the past (before the implementation of Continuous practices), which indicates an improved speed of delivery. This is also confirmed by Rodríguez et al. (2017), which saying that: "*Although similarities and differences exist among the emerging models of Continuous Deployment, three major themes characterize the models: (1) deployment, (2) continuity and (3) speed. Hence, continuous deployment means the ability to bring valuable product features to customers on demand and at will (deployment), in series or patterns with the aim of achieving continuous flow (continuity) and in significantl*y shorter cycles than traditional lead-times, from a couple of weeks, to days or even hours (speed)."

Performance improvements concerning improved quality were not observed. However, the potential CSF: *"Quality"* results in performance improvements and could be measured in the way of the degree

of test coverage of the code, to preserve quality coding. It should be noted that the performance improvement of this potential CSF which implies speed does not relate to this measurement.

The following potential CSF's: *"Goals"* and *"Strategy and approach"* may be measured, but does not result in performance improvements (criteria 2). These potential CSF's could be measured in the sense of the presence of documents.

All these measurements should imply that also the third criteria in order to be a verified CSF, according to Markus & Tanis (2000) that the success and performance of a CSF may be measured, has been met. In vain, these measurements are only indications of how the potential CSF's could be measured, but no measurement reports or documents showing that they measured the performance improvement have been provided.

This means that no verified CSF's were found, which met all criteria in order to be a verified CSF.

5.3. Recommendations for practice

The potential CSF's which have verified performance improvements may be carefully used when (new) DevOps teams want to adopt and implement Continuous practices, to indicate where the focus should be to improving the speed of delivery or achieving more story points per sprint.

5.4. Recommendations for further research

This research was conducted by one researcher, in one case organization. To strengthen the validity of the list with potential CSF's, more research in different case organizations is needed.

In this research only "T-shaped Programmers" were questioned. Other traditional DevOps roles like: "Testers", "Functional designers" or "Administrators" have not been questioned. Further research with different DevOps roles is needed to exclude that the perception of a person with a T-shaped DevOps role is reflecting the same viewpoints as someone else with a different T-shaped DevOps role, e.g. like a "T-shaped Tester". In order to check if there is missing evidence of performance improvements or ways or measurement of certain potential CFS's.

Further research is recommended in order to verify the remaining criteria on the list of potential CSF's, which relates to the main research question: "Which Critical Success Factors apply to the adoption and implementation of Continuous Practices in a DevOps context?"

The following potential CSF's are missing evidence of a performance improvement: *"Preconditions", "Goals", "Strategy and approach", "Process design", "Motivation", "Complexity across customer organization boundary", "Acceptance by customer", "Sales and intermediaries", "Coordination", "Communication", "Knowledge and training", "Tooling", "Pressure"*

The following potential CSF's are missing evidence of a way of measurement: *"Preconditions", "Goals", "Strategy and approach", "Architecture", "Process design", "Motivation", "Resistance to change", "Complexity across customer organization boundary", "Acceptance by customer", "Sales and intermediaries", "Quality", "Customer involvement", "Test complexity & source code control", "Coordination", "Communication", "Knowledge and training", "Tooling", "Pace", "Pressure"*.

References

Austel, P., Chen, H., Mikalsen, T., Rouvellou, I., Sharma, U., Silva-Lepe, I., & Subramanian, R. (2015). Continuous Delivery of Composite Solutions: A Case for Collaborative Software Defined PaaS Environments. Proceedings of the 2Nd International Workshop on Software-Defined Ecosystems, 3–6. https://doi.org/10.1145/2756594.2756595

Brandao Gomes da Silva, A. C., Carneiro, G. de F., Brito e Abreu, F., & Monteiro, M. P. (2017). Frequent Releases in Open Source Software: A Systematic Review. INFORMATION, 8(3). https://doi.org/10.3390/info8030109

Chen, L. (2015a). Continuous Delivery: Huge Benefits, but Challenges Too. *IEEE Software, 32*(2), 50-54. doi:10.1109/MS.2015.27

Chen, L. (2015b). Towards Architecting for Continuous Delivery. 2015 12th Working IEEE/IFIP Conference on Software Architecture, 131–134. https://doi.org/10.1109/WICSA.2015.23

Chen, L. (2017). Continuous Delivery: Overcoming adoption challenges. JOURNAL OF SYSTEMS AND SOFTWARE, 128, 72–86. https://doi.org/10.1016/j.jss.2017.02.013

Chen, L. (2018). Continuous Delivery at Scale: Challenges and Opportunities. Proceedings of the 4th International Workshop on Rapid Continuous Software Engineering, 42–42. https://doi.org/10.1145/3194760.3194764

Claps, G. G., Svensson, R. B., & Aurum, A. (2015). On the journey to continuous deployment: Technical and social challenges along the way. INFORMATION AND SOFTWARE TECHNOLOGY, 57, 21–31. https://doi.org/10.1016/j.infsof.2014.07.009

Clarke, P. M., Elger, P., & O'Connor, R. V. (2016). Technology Enabled Continuous Software Development. 2016 IEEE/ACM International Workshop on Continuous Software Evolution and Delivery (CSED), 48–48. https://doi.org/10.1109/CSED.2016.017

Colomo-Palacios, R., Fernandes, E., Soto-Acosta, P., & Larrucea, X. (2018). A case analysis of enabling continuous software deployment through knowledge management. *International Journal of Information Management, 40*, 186-189. doi:https://doi.org/10.1016/j.ijinfomgt.2017.11.005

de Jong, M., & van Deursen, A. (2015). Continuous Deployment and Schema Evolution in SQL Databases. Proceedings of the Third International Workshop on Release Engineering, 16–19. http://dl.acm.org/citation.cfm?id=2820690.2820699

Debroy, V., Miller, S., & Brimble, L. (2018). Building Lean Continuous Integration and Delivery Pipelines by Applying DevOps Principles: A Case Study at Varidesk. Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, 851–856. https://doi.org/10.1145/3236024.3275528

Dikert, K., Paasivaara, M., & Lassenius, C. (2016). Challenges and success factors for large-scale agile transformations: A systematic literature review. *Journal of Systems and Software, 119*, 87-108. doi:https://doi.org/10.1016/j.jss.2016.06.013

Dyck, A., Penners, R., & Lichter, H. (2015, 19-19 May 2015). *Towards Definitions for Release Engineering and DevOps.* Paper presented at the 2015 IEEE/ACM 3rd International Workshop on Release Engineering.

Eddy, B. P., Wilde, N., Cooper, N. A., Mishra, B., Gamboa, V. S., Patel, K. N., & Shah, K. M. (2017). CDEP: Continuous Delivery Educational Pipeline. Proceedings of the SouthEast Conference, 55–62. https://doi.org/10.1145/3077286.3077301

Erich, F. (2019). *DevOps is Simply Interaction Between Development and Operations.* In J.-M. Bruel, M. Mazzara, & B. Meyer (Eds.), Software Engineering Aspects of Continuous Development and New Paradigms of Software Production and Deployment (Vol. 11350, pp. 89-99). Cham: Springer International Publishing.

Esteves, J., & Pastor, J. (2003). *Using a multimethod approach to research enterprise systems implementations* (Vol. 2).

Esteves, J., & Pastor, J. A. (2006). *Organizational and Technological Critical Success Factors Behavior along the ERP Implementation Phases.* Paper presented at the Enterprise Information Systems VI, Dordrecht.

Esteves, J., & Pastor-Collado, J. (2000). *Towards the Unification of Critical Success Factors for ERP Implementations*.

Fabijan, A., Olsson, H. H., & Bosch, J. (2016). Time to Say “Good Bye”: Feature Lifecycle. 2016 42th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), 9–16. https://doi.org/10.1109/SEAA.2016.59

Fitzgerald, B., & Stol, K.-J. (2014). *Continuous Software Engineering and Beyond: Trends and Challenges*.

Fitzgerald, B., & Stol, K.-J. (2017). Continuous software engineering: A roadmap and agenda. *Journal of Systems and Software, 123*, 176-189. doi:https://doi.org/10.1016/j.jss.2015.06.063

Greising, L., Bartel, A., & Hagel, G. (2018). Introducing a Deployment Pipeline for Continuous Delivery in a Software Architecture Course. Proceedings of the 3rd European Conference of Software Engineering Education, 102–107. https://doi.org/10.1145/3209087.3209091

Gupta, V., Kapur, P. K., & Kumar, D. (2017). Modeling and measuring attributes influencing DevOps implementation in an enterprise using structural equation modeling. *Information and Software Technology, 92*, 75-91. doi:https://doi.org/10.1016/j.infsof.2017.07.010

Holland, C. R., & Light, B. (1999). A critical success factors model for ERP implementation. *IEEE Software, 16*(3), 30-36. doi:10.1109/52.765784

Humble, J., & Farley, D. (2010). Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation (Adobe Reader): Pearson Education.

Itkonen, J., Udd, R., Lassenius, C., & Lehtonen, T. (2016). *Perceived Benefits of Adopting Continuous Delivery Practices*.

Kärpänoja, P., Virtanen, A., Lehtonen, T., & Mikkonen, T. (2016). Exploring Peopleware in Continuous Delivery. Proceedings of the Scientific Workshop Proceedings of XP2016, 13:1–13:5. https://doi.org/10.1145/2962695.2962708

Karvonen, T., Behutiye, W., Oivo, M., & Kuvaja, P. (2017). Systematic literature review on the impacts of agile release engineering practices. INFORMATION AND SOFTWARE TECHNOLOGY, 86, 87–100. https://doi.org/10.1016/j.infsof.2017.01.009

Kitchenham, B. (2004). Procedures for Performing Systematic Reviews (Vol. 33).

Laukkanen, E., Lehtinen, T. O. A., Itkonen, J., Paasivaara, M., & Lassenius, C. (2016). Bottom-up Adoption of Continuous Delivery in a Stage-Gate Managed Software Organization. Proceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, 45:1–45:10. https://doi.org/10.1145/2961111.2962608

Laukkanen, E., Itkonen, J., & Lassenius, C. (2017). Problems, causes and solutions when adopting continuous delivery—A systematic literature review. *Information and Software Technology, 82*, 55-79. doi:10.1016/j.infsof.2016.10.001

Laukkanen, E., Paasivaara, M., Itkonen, J., & Lassenius, C. (2018). Comparison of release engineering practices in a large mature company and a startup. Empirical Software Engineering, 23(6), 3535–3577. https://doi.org/10.1007/s10664-018-9616-7

Leidecker, J. K., & Bruno, A. V. (1984). Identifying and using critical success factors. *Long Range Planning, 17*(1), 23-32. doi:https://doi.org/10.1016/0024-6301(84)90163-8

Leppanen, M., Makinen, S., Pagels, M., Eloranta, V.-P., Itkonen, J., Mantyla, M. V., & Mannisto, T. (2015). The highways and country roads to continuous deployment. *IEEE Software, 32*(2), 64-72. doi:10.1109/MS.2015.50

Lwakatare, L. E., Kuvaja, P., & Oivo, M. (2015). *Dimensions of DevOps.* Paper presented at the Agile Processes in Software Engineering and Extreme Programming, Cham.

Lwakatare, L. E., Kuvaja, P., & Oivo, M. (2016a). *An Exploratory Study of DevOps: Extending the Dimensions of DevOps with Practices*.

Lwakatare, L. E., Kuvaja, P., & Oivo, M. (2016b). *Relationship of DevOps to Agile, Lean and Continuous Deployment.* Paper presented at the Product-Focused Software Process Improvement, Cham.

Makinen, S., Leppanen, M., Kilamo, T., Mattila, A.-L., Laukkanen, E., Pagels, M., & Mannisto, T. (2016). Improving the delivery cycle: A multiple-case study of the toolchains in Finnish software

intensive enterprises. INFORMATION AND SOFTWARE TECHNOLOGY, 80, 175–194. https://doi.org/10.1016/j.infsof.2016.09.001

Mäntylä, M. V., Adams, B., Khomh, F., Engström, E., & Petersen, K. (2015). On rapid releases and software testing: A case study and a semi-systematic literature review. Empirical Software Engineering, 20(5), 1384–1425. https://doi.org/10.1007/s10664-014-9338-4

Mark Saunders, Philip Lewis, Adrian Thornhill. Research methods for business students, 7th edition, Pearson, 2016.

Markus, M., & Tanis, C. (2000). *The enterprise system experience—from adoption to success*.

Markus, M. L., Axline, S., Petrie, D., & Tanis, S. C. (2000). Learning from adopters' experiences with ERP: problems encountered and success achieved. *Journal of Information Technology (Routledge, Ltd.), 15*(4), 245-265. doi:10.1177/026839620001500402

Olsson, H. H., Alahyari, H., & Bosch, J. (2012). Climbing the "Stairway to Heaven"—A Mulitiple-Case Study Exploring Barriers in the Transition from Agile Development towards Continuous Deployment of Software. 2012 38th Euromicro Conference on Software Engineering and Advanced Applications, 392–399. https://doi.org/10.1109/SEAA.2012.54

Qumer Gill, A., Loumish, A., Riyat, I., & Han, S. (2017). DevOps for information management systems. *VINE Journal of Information and Knowledge Management Systems, 48*(1), 122-139. doi:10.1108/VJIKMS-02-2017-0007

Ram, J., Corkindale, D., & Wu, M.-L. (2013). Implementation critical success factors (CSFs) for ERP: Do they contribute to implementation success and post-implementation performance? International Journal of Production Economics, 144(1), 157-174. doi:https://doi.org/10.1016/j.ijpe.2013.01.032

Riungu-Kalliosaari, L., Mäkinen, S., Lwakatare, L. E., Tiihonen, J., & Männistö, T. (2016). DevOps Adoption Benefits and Challenges in Practice: A Case Study. In P. Abrahamsson, A. Jedlitschka, A. Nguyen Duc, M. Felderer, S. Amasaki, & T. Mikkonen (Red.), Product-Focused Software Process Improvement (Vol. 10027, pp. 590–597). Springer International Publishing. https://doi.org/10.1007/978-3-319-49094-6_44

Rodríguez, P., Haghighatkhah, A., Lwakatare, L. E., Teppola, S., Suomalainen, T., Eskeli, J., . . . Oivo, M. (2017). Continuous deployment of software intensive products and services: A systematic mapping study. *The Journal of Systems & Software, 123*, 263-291. doi:10.1016/j.jss.2015.12.015

Savor, T., Douglas, M., Gentili, M., Williams, L., Beck, K., & Stumm, M. (2016). Continuous Deployment at Facebook and OANDA. Proceedings of the 38th International Conference on Software Engineering Companion, 21–30. https://doi.org/10.1145/2889160.2889223

Schermann, G., Schöni, D., Leitner, P., & Gall, H. C. (2016). Bifrost: Supporting Continuous Deployment with Automated Enactment of Multi-Phase Live Testing Strategies. Proceedings of the 17th International Middleware Conference, 12:1–12:14. https://doi.org/10.1145/2988336.2988348

Shahin, M., Babar, M. A., & Zhu, L. (2016). The Intersection of Continuous Deployment and Architecting Process: Practitioners' Perspectives. Proceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, 44:1–44:10. https://doi.org/10.1145/2961111.2962587

Shahin, M., Babar, M. A., & Zhu, L. (2017). Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices. IEEE ACCESS, 5, 3909–3943. https://doi.org/10.1109/ACCESS.2017.2685629

Shahin, M., Zahedi, M., Babar, M. A., & Zhu, L. (2017b). Adopting Continuous Delivery and Deployment: Impacts on Team Structures, Collaboration and Responsibilities. Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering, 384–393. https://doi.org/10.1145/3084226.3084263

Shahin, M., Babar, M. A., Zahedi, M., & Zhu, L. (2017c). Beyond Continuous Delivery: An Empirical Investigation of Continuous Deployment Challenges. Proceedings of the 11th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, 111–120. https://doi.org/10.1109/ESEM.2017.18

Shahin, M., Zahedi, M., Babar, M. A., & Zhu, L. (2018). An empirical study of architecting for continuous delivery and deployment. Empirical Software Engineering. https://doi.org/10.1007/s10664-018-9651-4

Shahzeydi, M., Gandomani, T. J., & Sadeghi, R. (2018). Quality Aspects of Continuous Delivery in Practice. INTERNATIONAL JOURNAL OF ADVANCED COMPUTER SCIENCE AND APPLICATIONS, 9(5), 210–212.

Siqueira, R., Camarinha, D., Wen, M., Meirelles, P., & Kon, F. (2018). Continuous delivery: Building trust in a large-scale, complex government organization. IEEE Software, 35(2), 38–43. https://doi.org/10.1109/MS.2018.111095426

Skelton, M., & O'Dell, C. (2016). Continuous Delivery with Windows and .NET. Champaign, Verenigde Staten: O'Reilly Media, Inc

Smeds, J., Nybom, K., & Porres, I. (2015, 2015//). *DevOps: A Definition and Perceived Adoption Impediments.* Paper presented at the Agile Processes in Software Engineering and Extreme Programming, Cham.

Ståhl, D., & Bosch, J. (2014). Modeling continuous integration practice differences in industry software development. *Journal of Systems and Software, 87*, 48-59. doi:https://doi.org/10.1016/j.jss.2013.08.032

Stahl, D., Martensson, T., & Bosch, J. (2017, 30 Aug.-1 Sept. 2017). *Continuous practices and devops: beyond the buzz, what does it all mean?* Paper presented at the 2017 43rd Euromicro Conference on Software Engineering and Advanced Applications (SEAA).

Ståhl, D., Mårtensson, T., & Bosch, J. (2017). The continuity of continuous integration: Correlations and consequences. *Journal of Systems and Software, 127*, 150-167. doi:https://doi.org/10.1016/j.jss.2017.02.003

Strauss, A., & Corbin, J. (1998). Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory. Thousand Oaks, CA: Sage Publications, Inc.

Toh, M. Z., Sahibuddin, S., & Mahrin, M. N. r. (2019). Adoption Issues in DevOps from the Perspective of Continuous Delivery Pipeline.

Vassallo, C., Zampetti, F., Romano, D., Beller, M., Panichella, A., Penta, M. D., & Zaidman, A. (2016). Continuous Delivery Practices in a Large Financial Organization. 2016 IEEE International Conference on Software Maintenance and Evolution (ICSME), 519–528. https://doi.org/10.1109/ICSME.2016.72

Virtanen, A., Kuusinen, K., Leppänen, M., Luoto, A., Kilamo, T., & Mikkonen, T. (2017). On Continuous Deployment Maturity in Customer Projects. Proceedings of the Symposium on Applied Computing, 1205–1212. https://doi.org/10.1145/3019612.3019777

Weber, I., Nepal, S., & Zhu, L. (2016). Developing Dependable and Secure Cloud Applications. *IEEE Internet Computing, 20*(3), 74-79. doi:10.1109/MIC.2016.67

Wettinger, J., Andrikopoulos, V., & Leymann, F. (2015a). Enabling DevOps Collaboration and Continuous Delivery Using Diverse Application Environments. Paper presented at the On the Move to Meaningful Internet Systems: OTM 2015 Conferences, Cham.

Wettinger, J., Breitenbucher, U., & Leymann, F. (2015b). Dyn Tail - Dynamically Tailored Deployment Engines for Cloud Applications. Paper presented at the 2015 IEEE 8th International Conference on Cloud Computing (CLOUD).

Wettinger, J., Breitenbücher, U., & Leymann, F. (2014). DevOpSlang – Bridging the Gap between Development and Operations. In C. Salinesi, M. C. Norrie, & Ó. Pastor (Eds.), Advanced Information Systems Engineering (Vol. 7908, pp. 108-122). Berlin, Heidelberg: Springer Berlin Heidelberg.

Yaman, S. G., Sauvola, T., Riungu-Kalliosaari, L., Hokkanen, L., Kuvaja, P., Oivo, M., & Männistö, T. (2016). Customer Involvement in Continuous Deployment: A Systematic Literature Review. In M. Daneva & O. Pastor (Red.), Requirements Engineering: Foundation for Software Quality (Vol. 9619, pp. 249–265). https://doi.org/10.1007/978-3-319-30282-9_18

R.K. Yin. Case Study Research: Design and Methods Third Edition Applied Social Research Methods Series Vol 5, volume 5. Sage Publications, Incorporated, 3rd edition, December 2008.

Yin., R.K. (2014). Case Study Research: Design and Methods. 5th edition. SAGE Publications.

Appendices

The following files belong to this study: Appendix A{x}-Z{x}. Files can be requested from the author.

Appendix A

File: Appendix A.xlsx

Appendix AA

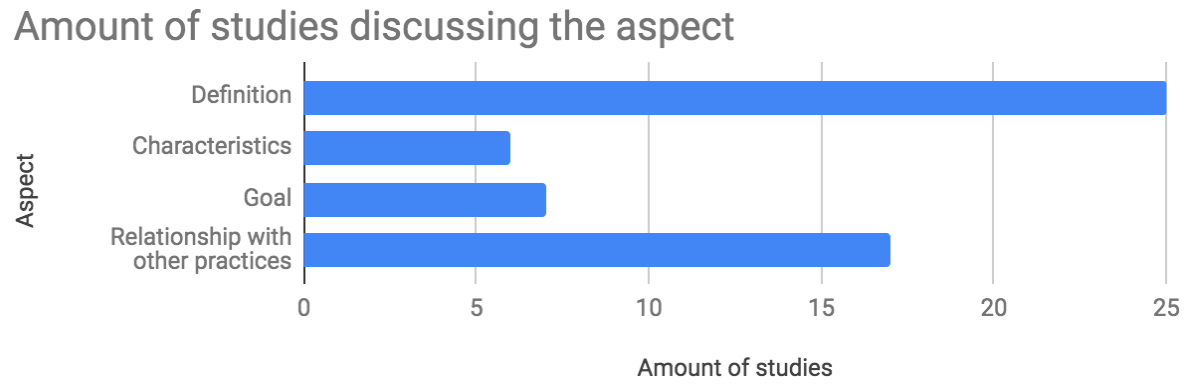

Chart 1 - Amount of studies discussing aspects of frequent/rapid or continuous delivery/deployment or releasing of software (features) in this SLR (N = 40)

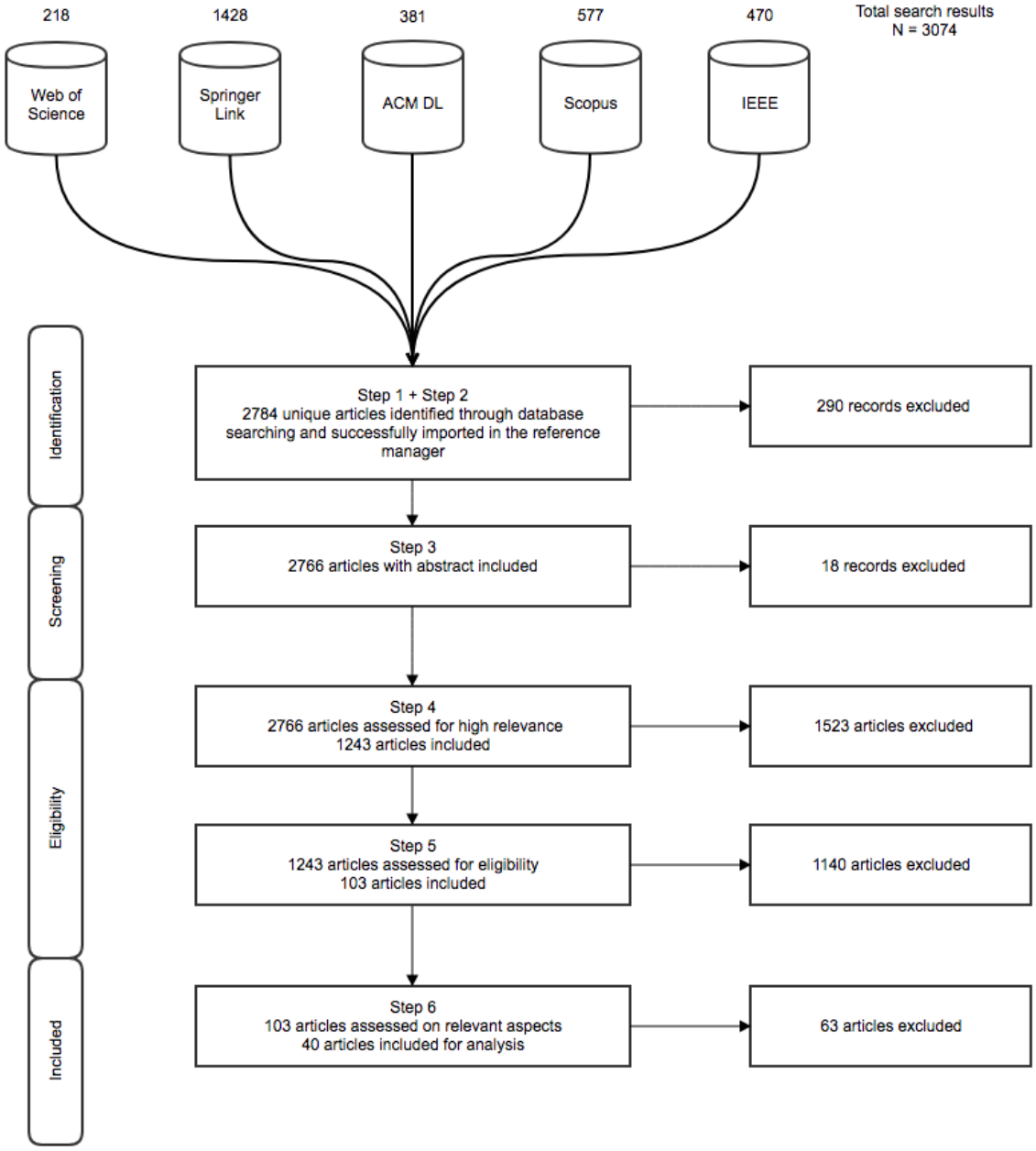

Figure 3 - Systematic Literature Review results for SQ1

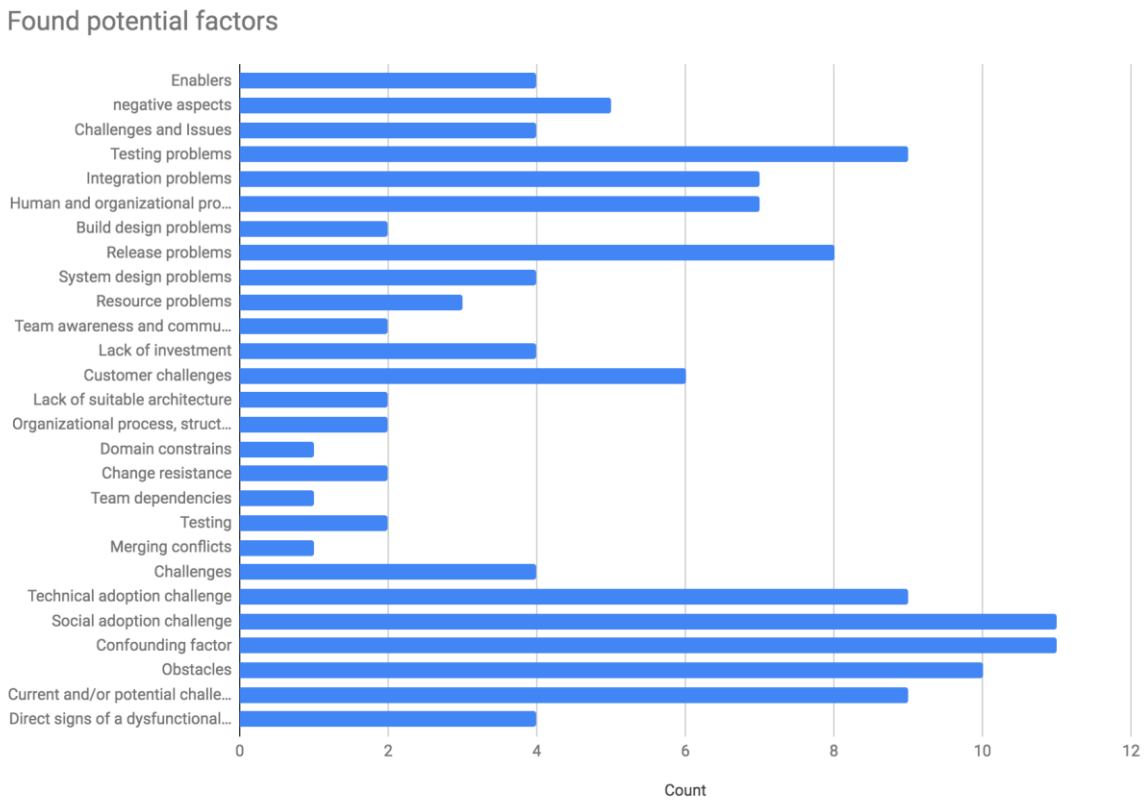

Chart 2 - Amount of studies discussing aspects which are barring/delaying frequent/rapid or continuous delivery/deployment or releasing of software (features) in this SLR (N = 14)

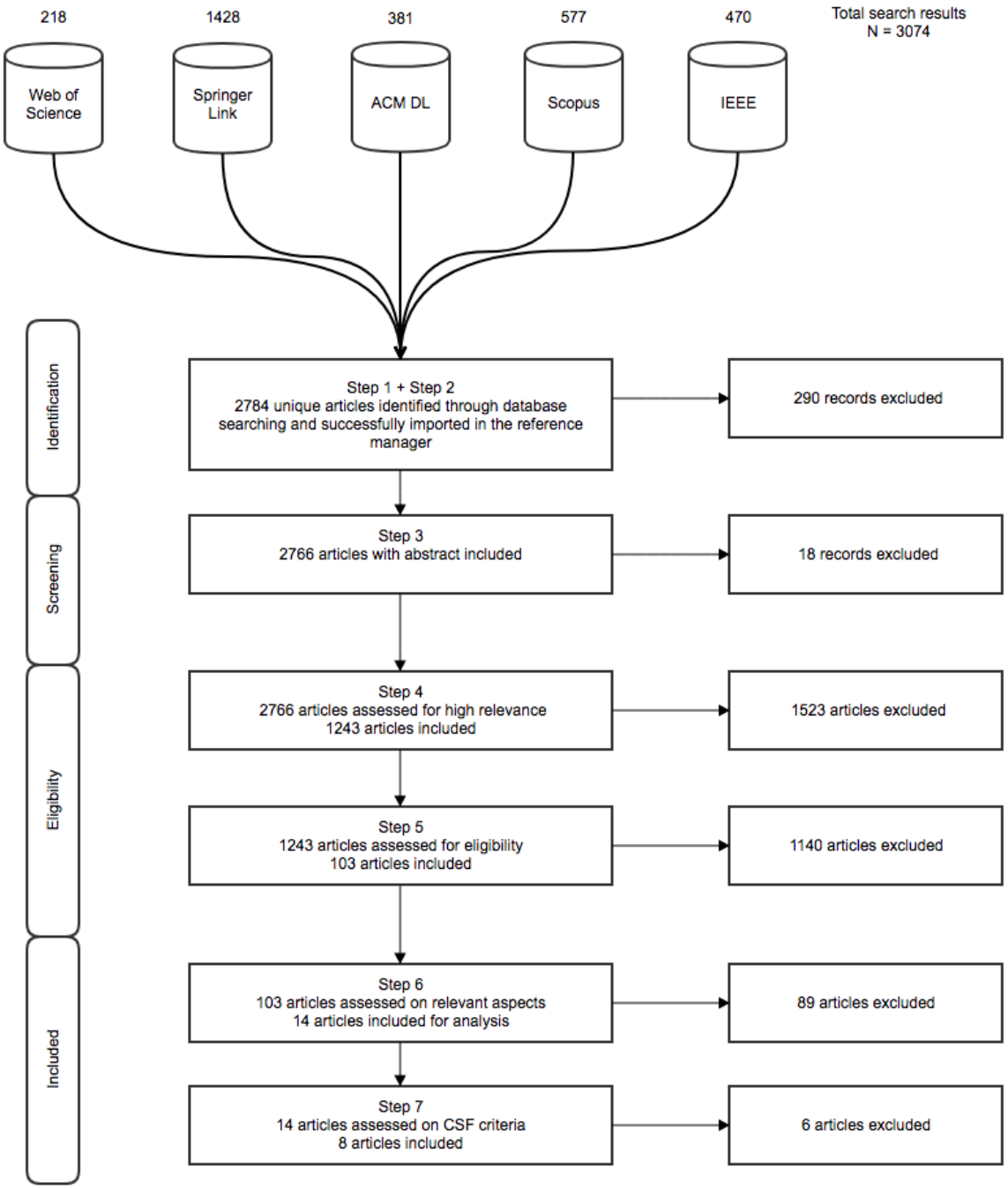

Figure 4 - Systematic Literature Review results for SQ2

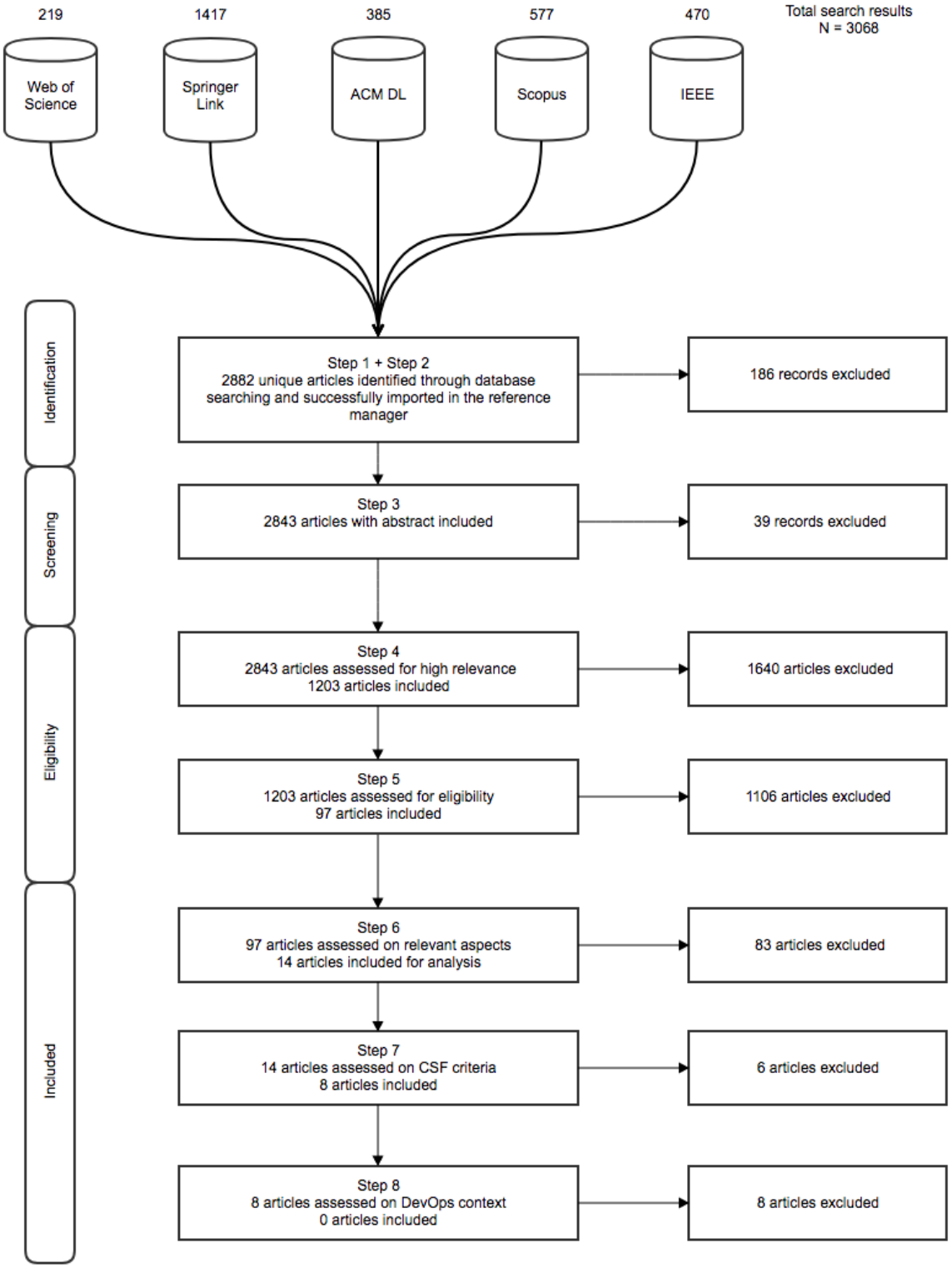

Figure 5 - Systematic Literature Review results for SQ3

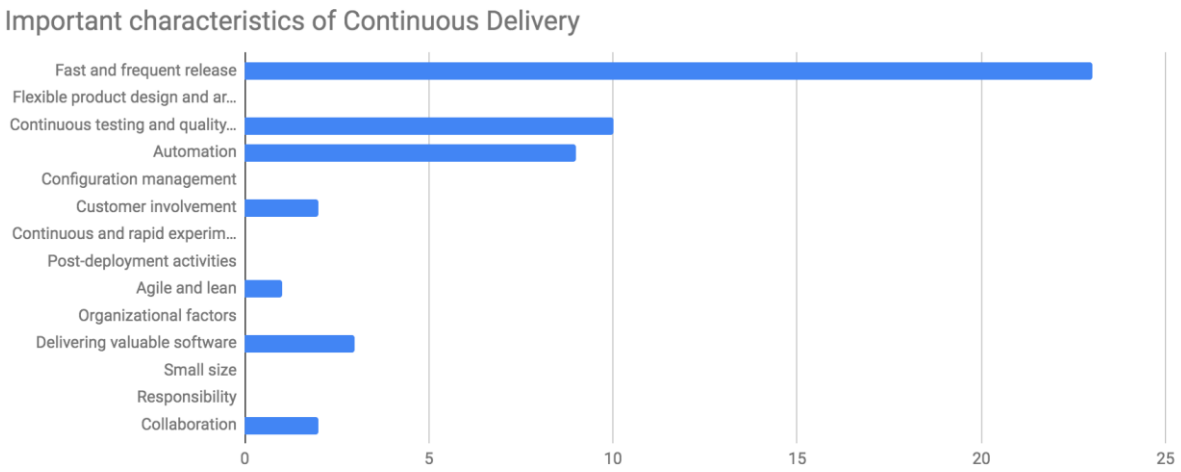

Chart 3 - Results of cross-checked meanings / definitions against characteristics of Continuous Delivery (total=31)

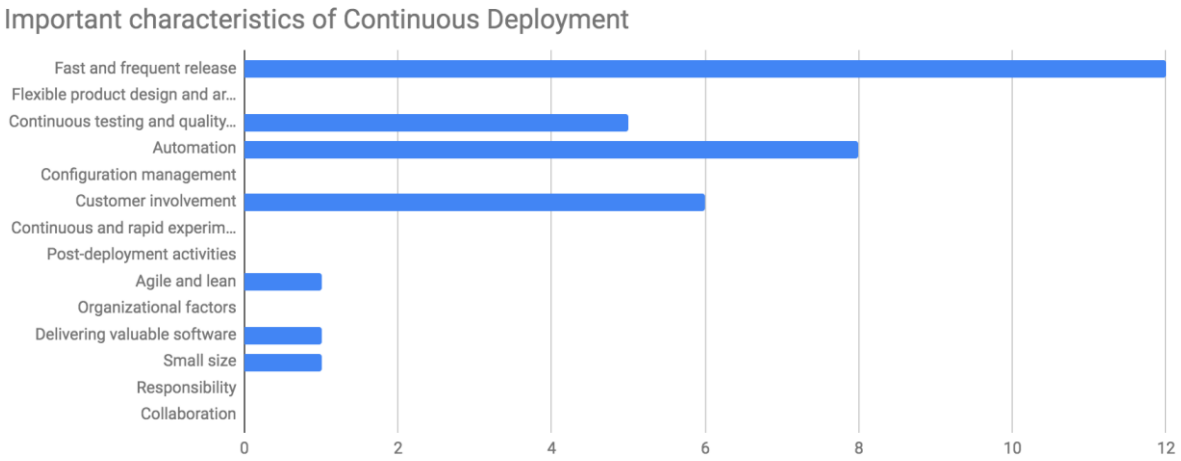

Chart 4 - Results of cross-checked meanings / definitions against characteristics of Continuous Deployment (total=17)

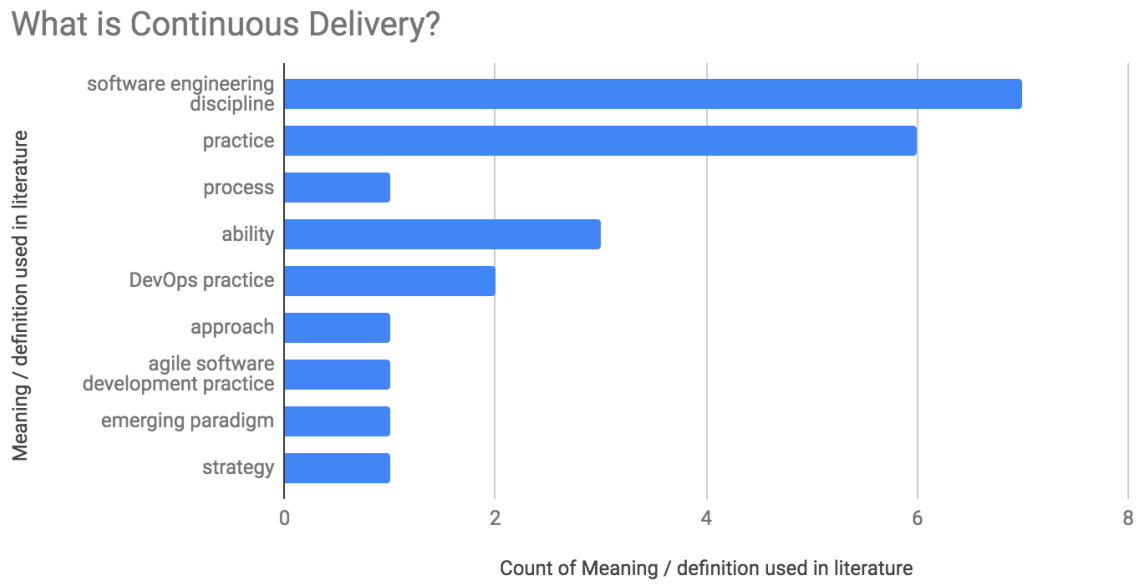

Chart 5 - Results of what Continuous Delivery stands for

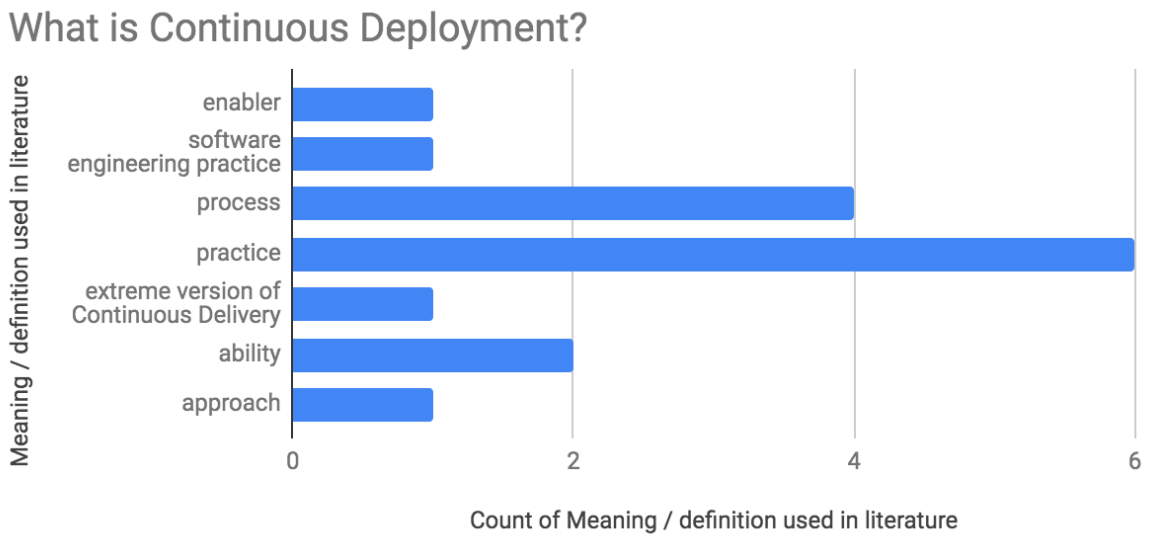

Chart 6 - Results of what Continuous Deployment stands for

Table 2 - Found characteristics of Continuous Delivery and Continuous Deployment

Name		**Description**	**Study**
Fast and frequent release		Ability to release software whenever the organization wants to (on demand or at will) based on need and with preference given to shorter cycles or even continuous flow (weekly or daily).	Rodriguez et al. (2017); Chen (2015b)
Flexible product design and architecture		CD requires evolutionary and robust software architecture with the aim of balancing speed and stability.	Rodriguez et al. (2017)
Continuous testing and quality assurance		Ensuring the quality of the software at all times without compromises despite the need for fast and continuous deployment.	Rodriguez et al. (2017); Laukkanen et al. (2017); Schermann et al. (2016)
Automation		Automating the delivery pipeline from building and testing to deployment and monitoring. (Rodriguez et al., 2017) Tools enabling easy automatic deployment and testing (Mäntylä et al., 2015) The CD practice is achieved with discipline and by automating the delivery, including building, testing and deploying the software (Laukkanen et al., 2016)	Rodriguez et al. (2017); Laukkanen et al. (2016); Laukkanen et al. (2017); Chen (2015b); Mäntylä et al. (2015)
Configuration management		Version control branching strategies and system configuration management approaches to enable CD.	Rodriguez et al. (2017)
Customer involvement		Mechanisms to involve customers in the development process and collect customer feedback from deliveries as early as possible (even near real-time) to drive design decisions and innovation. (Rodriguez et al., 2017) In general, customer input	Rodriguez et al. (2017); Yaman et al. (2016); Mäntylä et al. (2015);

		enhances CD activities and benefits both developers and end users. (Yaman et al., 2016) The study of Mäntylä et al. (2015) consider the involvement of customers as enabler of Rapid Releases (RR).	
Continuous and rapid experimentation		Systematical design and execution of small field experiments to guide product development and accelerate innovation.	Rodriguez et al. (2017)
Post-deployment activities		Activities that are conducted once the product (or a new feature or enhancement of the product) has been deployed to support fast business and technical decision making.	Rodriguez et al. (2017)
Agile and lean		Extending agile and lean software development towards continuous flow to support CD.	Rodriguez et al. (2017)
Organizational factors		Organizational factors that enable CD (integrated corporative functions, transparency and innovative and experimental organizational culture).	Rodriguez et al. (2017)
Delivering valuable software		The team continuously makes sure the software being developed provides value to customers. This implies the ability to quickly gather users' feedback on a feature once it is delivered.	Chen (2015b)
Small size		A big feature will be decomposed into smaller ones, so that each one can be quickly delivered to get feedback early.	Chen (2015b)
Responsibility		Implementing CD practices demands skill-set and knowledge that are either brand new (e.g., tools for automating CD process), or lie at the intersection of development and operations responsibilities.	Karpanoja et al. (2016); Shahin et al. (2017b)
Collaboration		Continuous Delivery and DevOps have emerged with the goal to bring together developers and operations personnel by enabling their efficient collaboration.	Lwakatare et al. (2016b); Wettinger et al. (2015a); Makinen et al. (2016)

Table 3 - Found potential Critical Success Factors (92) and potential prerequisite Critical Success Factors (34)

Potential CSF Name	**Pre**	**Description**	**Study**
Ambiguous test result		Test result is not communicated to developers, is not an explicit pass or fail or it is not clear what broke the build.	Laukkanen et al. (2017)
Changing roles		Different roles need to adapt for collaboration. / Team members have to adapt to different tasks in new roles due to working in a CD environment	Laukkanen et al. (2017); Claps et al. (2015)
Complex build		Build system, process or scripts are complicated or complex.	Laukkanen et al. (2017)
Complex testing		Testing is complex, e.g., setting up environment.	Laukkanen et al. (2017)

Customer data preservation		Preserving customer data between upgrades.	Laukkanen et al. (2017)
Deployment downtime		Downtime cannot be tolerated with frequent releases.	Laukkanen et al. (2017)
Documentation		Keeping the documentation in-sync with the released version.	Laukkanen et al. (2017); Claps et al. (2015)
Effort		Initially setting up continuous delivery requires effort.	Laukkanen et al. (2017)
Flaky tests		Tests that randomly fail sometimes.	Laukkanen et al. (2017); Laukkanen et al. (2016)
Hardware testing		Testing with special hardware that is under development or not always available.	Laukkanen et al. (2017)
Inflexible build		The build system cannot be modified flexibly.	Laukkanen et al. (2017)
Insufficient hardware resources		Build and test environments require hardware resources.	Laukkanen et al. (2017)
Internal dependencies		Dependencies between parts of the software system.	Laukkanen et al. (2017)
Lack of discipline		Discipline to commit often, test diligently, monitor the build status and fix problems as a team.	Laukkanen et al. (2017)
Lack of experience		Lack of experience practicing CI or CD. CI, CDE, and CD demand new technical and soft skills Need highly skilled developers / Having an experienced team is critical in the successful adoption of CD	Laukkanen et al. (2017); Shahin et al. (2017); Claps et al. (2015)
Lack of motivation		People need to be motivated to get past early difficulties and effort.	Laukkanen et al. (2017)
More deployed bugs		Frequent releases cause more deployed bugs.	Laukkanen et al. (2017)
More pressure		Increased amount of pressure because software needs to be in always-releasable state.	Laukkanen et al. (2017)
Multi-platform testing		Testing with multiple platforms when developers do not have access to all of them.	Laukkanen et al. (2017)
Network latencies		Network latencies hinder continuous integration.	Laukkanen et al. (2017)
Organizational structure		Organizational structure, e.g., separation between divisions causes problems.	Laukkanen et al. (2017)
Problematic deployment		Deployment of the software is time-consuming or error-prone.	Laukkanen et al. (2017)
System modularization		The system consists of multiple units, e.g., modules or services.	Laukkanen et al. (2017)
Team coordination		Increased need for team coordination.	Laukkanen et al. (2017)
Third party integration		Frequent releases complicate third party integration.	Laukkanen et al. (2017)
Time-consuming testing		Testing takes too much time.	Laukkanen et al. (2017)
UI testing		Testing the UI of the application.	Laukkanen et al. (2017)
Unsuitable architecture		System architecture limits continuous delivery.	Laukkanen et al. (2017)
Untestable code		Software is in a state that it cannot be tested.	Laukkanen et al. (2017)

Users do not like updates		Users might not like frequent updates.	Laukkanen et al. (2017); Shahin et al. (2017); Claps et al. (2015); Shahin et al. (2017c)
Feature discovery		Users might not discover new features.	Laukkanen et al. (2017); Yaman et al. (2016); Claps et al. (2015)
Marketing		Marketing versionless system.	Laukkanen et al. (2017)
Coordination and collaboration challenges		Practicing CI, CDE, CD needs more and effective coordination and communication between team members	Shahin et al. (2017)
Cost		Major upgrade in infrastructures and resources Training and coaching	Shahin et al. (2017)
Customer environment		Lack of access to customer environment	Shahin et al. (2017); Virtanen et al. (2017)
Customer environment		Complex and manual configuration	Shahin et al. (2017)
Customer environment		Diversity and complexity of customer sites	Shahin et al. (2017)
Customer environment		Difficulty to stimulate production-like environment	Shahin et al. (2017)
Dependencies in design and code		Highly coupled architectures Difficulty to find autonomous requirements for frequent integrations	Shahin et al. (2017)
Dependencies with hardware and other (legacy) applications		Releasing an application on continuous basis requires deploying all dependent applications in customer site Hardware and network dependencies	Shahin et al. (2017)
Difficulty to change established organizational polices and cultures		Lack of agile and suitable business model Changing long-lived feature branching to short-lived one in an established company	Shahin et al. (2017)
Distributed organization		Distributed team model Inconsistent perceptions among team members	Shahin et al. (2017)
Domain constraints		Some domains don't allow or cause difficulties to truly adopt and implement CD / Whilst CD practice is more easily applied to web-based applications, it may not be easily applied to other domains such as embedded systems and financial systems [9]. One possible reason for this is that such domains are more conservative to automatic deployment to customers and require more (manual) verifications before each release	Shahin et al. (2017); Shahin et al. (2017c)
General resistance to change		Changing the old habits of team members Time-consuming process to change team mindset	Shahin et al. (2017)
Lack of awareness and transparency		Lack of awareness and transparency in the delivery process Lack of understanding about the status of project increased number of merge conflicts	Shahin et al. (2017)
Lack of suitable tools and technologies		Lack of mature tools for automating tests and reviewing code in CI Frequency changes in tools Security and reliability issues in build and deployment tools Current tools don‘t fit to all organizations	Shahin et al. (2017)

More pressure and workload for team members		More stress for developers and operations team More responsibilities for developers	Shahin et al. (2017)
Scepticism and distrust on continuous practices		Lack of trust on benefits of CI, CDE, CD	Shahin et al. (2017)
Team dependencies		Cross-team dependencies Ripple effects of changes on multiple teams Dependency between feature team and module team in embedded system domain	Shahin et al. (2017)
Database schemas changes		Frequent changes in database schema	Shahin et al. (2017); Laukkanen et al. (2017); Claps et al. (2015)
Lack of proper test strategy		Lack of fully automated testing Lack of test-driven development	Shahin et al. (2017)
Poor test quality		Instable tests Low test coverage Low quality test data Long running tests Test dependencies	Shahin et al. (2017); Laukkanen et al. (2016)
Merging conflicts		Third party components Incompatibly among dependent components Lack of understanding about changed components	Shahin et al. (2017)
Customer unwillingness		Customers are reluctant to accept new functionalities mainly because of poor quality of releases.	Rodriguez et al. (2017)
Increased QA effort		Difficulties in managing the test automation infrastructure.	Rodriguez et al. (2017)
Company-wide effort		a lack of motivation for adopting CD	Claps et al. (2015)
Deploying the product		adopting the practice of small batches	Claps et al. (2015)
Infrastructure		requires proper hardware and software to handle the CD process and its related problems e.g. cross-product dependencies	Claps et al. (2015)
Partner plugins		only a small fraction of plugins are available due to code integration issues involved with a continuously changing software product	Claps et al. (2015)
Pressure		software developers may feel an increased amount of pressure to have code ready to be deployed immediately	Claps et al. (2015)
Process documentation		a lack of understanding of the CD process by novice developers due to inconsistent documentation and a lack of industry standards	Claps et al. (2015)
Product marketing		marketing CD products requires marketing a versionless product, which requires alternative marketing strategies	Claps et al. (2015)
Seamless upgrades		complications in implementation of seamless upgrades due to resource limitations, zero downtime deployment and customer data preservation	Claps et al. (2015)
Shorten customer feedback		Being able to analyse data when justifying the deployment of a new feature	Claps et al. (2015)

Source code control		having one single branch of code to maintain one working version of the software product in a CD environment	Claps et al. (2015)
Team coordination		requires extra effort to coordinate multiple teams	Claps et al. (2015)
Testing		having production quality tests and maintaining the process of 'code review'	Claps et al. (2015)
Product quality		the quality of the software product may decrease since bugs may slip through and be deployed to customers since deployments occur more frequently	Claps et al. (2015)
Lack of fully automated user acceptance test		Automating existing manual tests	Shahin et al. (2017c)
Lack of fully automated user acceptance test - Too much effort with low gain		Concerns about the potential benefits of automating acceptance test at large scale compared to its associated complexities and costs	Shahin et al. (2017c)
Lack of fully automated user acceptance test - Tools limitations		A few of the survey participants reported that current tools and technologies are immature to fully automate acceptance tests	Shahin et al. (2017c)
Lack of fully automated user acceptance test - Lack of automation skills		Lack of automation skills was another reason for having manual acceptance testing.	Shahin et al. (2017c)
Manual quality check		The most common manual task mentioned by the interviewees was code review. The interviewees' organizations performed several manual code reviews before deploying software to production. This is partially because some organizations may not have highly skilled developers for truly practicing CD.	Shahin et al. (2017c)
Deployment as business decision		The interviewees indicated that in their organizations the development teams were not able to immediately deploy every change to production environment, despite passing all the tests and quality checks. This is mainly because deploying to production was considered as a business decision, which had to be made by management or financial sectors. In other words, development team members have little control over production deployment. Furthermore, different organizations may adopt different policies and timeslots for release, which can bring the most value to their customers [3, 37].	Shahin et al. (2017c)
Insufficient level of automated test coverage		insufficient level of automated test coverage reduced the confidence of some interviewees' organizations in the readiness of their applications for actual deployment.	Shahin et al. (2017c)
Highly bureaucratic deployment process		a highly bureaucratic process as the one having a large number of formal tasks (e.g., getting approvals from various people) to be performed manually before each release.	Shahin et al. (2017c)
Lack of efficient rollback mechanism		Lack of efficient and automated rollback mechanism to quickly recover issues in deployment process may put software organizations at risk of delivering buggy code to their customers.	Shahin et al. (2017c)
Dependency at application level		It means organizations need to ensure that there is no integration problem when deploying an application to production. Deploying software changes on a continuous basis necessitates continuously deploying all dependencies (e.g., dependent applications).	Shahin et al. (2017c)
Customer environment		It was often stated by the interviewees that lack of carefully studying and exploring customer environments before moving to CD led to challenges in continuous and automatic release.	Shahin et al. (2017c)
Customer environment - Manual configuration of complex software		manual configuration of complex software, particularly when there is a tight coupling between software and hardware, and regulatory environments represented a significant obstacle to CD success.	Shahin et al. (2017c)

Customer environment - Hard to simulate/access real production		When there is no direct and regular access to customer environments, a software development team needs more communication with operations team at customer side to get confirmation and agreement for each release. / The interviewees frequently shared that it is not easy (if possible) to stimulate production-like environments with realistic data. Therefore, lack of access to and control on production environment make it much harder to fully automate deployment process.	Shahin et al. (2017c)
Manual interpretation of test results		With the increasing number of test cases, the interpretation of test results becomes quite time-consuming and labor-intensive process.	Shahin et al. (2017c)
Customer perception and behaviour		Customers' perceptions of their involvement in development are challenging. They might feel disturbed or interrupted from their on-going work (e.g. by product surveys or pop-up windows), and they might give negative or insufficient feedback. The customer can also be unsure about what they want and how to express it	Yaman et al. (2016)
Communication		Communication with the customer portrays difficulties, such as establishing trust before collaboration and choosing the right form of communication strategy. Also, managing the process, such as by dealing with conflicts, with different stakeholders is challenging	Yaman et al. (2016)
Data management		The customer-related data collection process and analysis of the data reveal several challenges. For example, the internal verification loop of the collected data has to be short and systematic, and feedback should be coming from the right channel. Similarly, the data analysis process requires high effort to, for example, work with data with noise in it, eliminate human factors such as subjectivity or prioritise tasks	Yaman et al. (2016)
Setting the scenes		Preparing and receiving the customer input is time-consuming. For example, creating detailed feedback might be challenging due to a lack of time, and organising workshops, questionnaires, interviews, site visits or personal interactions might be expensive and laborious. Also, different data collection techniques bring different difficulties; for example, a theatre session requires sophisticated technology at a special location. Sometimes it might be necessary to educate the customer about a common, helpful way of providing feedback	Yaman et al. (2016)
Transparency		Transparency in data, process and feedback affect users' intention to provide input. Limited or no transparency demotivates users to provide feedback. However, too much transparency causes customers to interfere with developers' work. Also, due to high visibility, failures might be too visible to customers	Yaman et al. (2016)
Customer profile		The needs of different user groups might diverge and change in different contexts. Establishing a customer sample group where all possible types of users are represented is challenging. Also, the customer's level of competence, experience, knowledge and/or reliability influences the success of customer involvement	Yaman et al. (2016)
Experiments and A/B testing		The customer might not want to be a part of an experiment or they might not welcome partially developed functionality. Moreover, conducting several experiments in parallel and interpreting the results are challenging. Determining where to start to experiment with the customer is another challenge	Yaman et al. (2016)
Sales and suppliers		Sometimes direct user data might not be accessible due to intermediaries, such as when a company does not sell products directly to end users. From the suppliers' point of view, they might not be interested in collecting customer feedback after selling a product or a service	Yaman et al. (2016)
Failing Builds		The case organization suffered from failing builds. The existence of build failures is not necessarily a problem for the CD practice, but they become problematic if they are common and not fixed immediately. It was reported that builds were failing for long time periods and sometimes people had to fix builds that others had broken. It was suggested that an automatic reverting system could solve the problem; if a change broke the build, then that change could be automatically reverted to keep the build unbroken.	Laukkanen et al. (2016)

Slow Feedback		The developers received some of the feedback for their changes slowly. For example, the whole product was not integrated all the time; only the changes that would go into a release were integrated together. In addition, limited hardware resources limited the possibility to do manual testing, which delayed the feedback.	Laukkanen et al. (2016)

Table 8 - Selected respondents

Alias	**DevOps team**	**Role**	**Current role experience**	**Current role experience in case organization**	**Total DevOps / Continuous practices experience**	**DevOps / Continuous practices experience in case organization**	**Attitude towards success of DevOps and implementation of Continuous practices**
R1	yes	Developer	15 years	14 years	3 years	3 years	Positive
R2	no	Software Architect / Lead DevOps Engineer	Since 2009 - 10 years	Since 2009 – 10 years	Since 2008 – 11 years	Since 2008 – 11 years	Positive
R3	yes	Senior Developer	14 years	4 years	7 years	3 years	Positive
R4	yes	Software Engineer	33 years	21 years	Approx. 5 years	Approx. 5 years	Neutral
R5	yes	Senior Developer / Technical Designer	15 years	15 years	3 years	3 years	Positive
R6	yes	Developer / Scrummaster	Developer 15 years; Scrummaster 5 years	7 years; 5 years	9 years	7 years	Neutral

Table 5 – Coding scheme (translated into English)

Relates to question	**Code**	**Sub-code**	**Definition**
2a	Role/T-profile		Description of the role / T-profile of the respondent
2b	DevOps team goal		Description of the shared goal

2c	Performance requirement	*{Name}*	Description of the performance requirement
2c	Measurable	*{Name}*	Description of the measurable unit of the performance requirement
2d	{CSF *Name}* -	Definition clear	Understanding of the CSF
2d	{CSF *Name}* -	Definition unclear	Misunderstanding of the CSF
2e	{CSF *Name}* -	Experienced	Experienced CSF by respondent
2e	{CSF *Name}* -	Not experienced	Un-experienced CSF by respondent
2e	{CSF *Name}* -	Example	Practical example of the CSF - (criteria1)
2f	{CSF *Name}* -	Weight	Weight of the CSF
2g	{CSF *Name}* -	Why?	Reason for weight indication of CSF
2e	{CSF *Name}* -	Why not?	Reason why the CSF is not experienced
	{CSF *Name}* -	Performance improvement	Description of the performance improvement - (criteria2)
2h*	{CSF *Name}* -	Measurable	Description of the measurable unit - (criteria3)
2i*	{CSF *Name}* -	Addressed	Description of how the CSF has been addressed
	Contextual relation -	*{CSF Name} - {Context}*	Description of the contextual relation between a potential CSF and a shared goal or performance requirement.
	Relationship -	Mutual interdependence	Description of mutual interdependence between DevOps and Continuous practices.
	Context information		A piece of description of the context, which can be used to support other codes.
1-8	Answer to selection criteria	{0-9}	Answers concerning the organization selection criteria
3	Answer to selection criteria	3 - {Name}	Answers concerning the organization selection criteria

Table 6 - Structure of the results concerning the organization selection criteria

Related selection criteria 1 + 2	**Description of the case organization**	Description of the case organization, background, IT-services, software development and amount of employees.

Related selection criteria 3	**DevOps context**	Description of the image, how DevOps is seen within the case organization, the amount of teams and how they collaborate, the size of teams, the variety of (traditional) roles, expertise and responsibilities.
Related selection criteria 4	**Implementation of Continuous Integration, Continuous Delivery and/or Deployment**	Description of the choice why there is chosen to implement Continuous Integration, Continuous Delivery and/or Continuous Deployment and how long ago this was successfully implemented.
Related selection criteria 5	**Delivery pipeline (process steps)**	Description of the delivery pipeline and the positions of roles.
Related selection criteria 5	**Roles / T-profiles**	Description of the variety of roles and T-profiles.
Related selection criteria 6	**Shared goals**	Description of the shared goals of the different DevOps teams.
Related selection criteria 7	**Performance requirements**	Description of the performance requirements of the different DevOps teams.
Related selection criteria 7	**Measurement of performance requirements**	Description of the measurements of the performance requirements of the different DevOps teams.
Related selection criteria 8	**Willing to submit interview candidates**	Brief conclusion whether the organization wants to participate in this study.
	Conclusion	Final conclusion, why the organization is selected as suitable case organization.

Table 7 - Structure of the results per CSF

Description of definition	A brief description of the CSF according to the list of potential CSF's.
Clear definition	A description of the synthesis of the respondents' responses, whether they found the definition of the CSF clear enough and were able to interpret the CSF correctly.
Experience	A description of the synthesis of the respondents' responses, whether they have ever experienced the CSF.
Examples	A description of one or more examples concerning the CSF, given by (a) respondent(s).

Weights	A table with the weights with substantiation given by the respondents concerning the CSF.
Performance improvements	A description of the performance improvement concerning the CSF.
Measurement	A description about how the CSF was measured.
Implementation	A description about how the CSF was addressed / implemented.
(Contextual) relationships	A description of the (contextual) relationship(s).
Conclusion	Concise description of the CSF positioned in the case organization and a reflection on the CSF if it can be seen as verified CSF.

Appendix B

File: Appendix B.xlsx

Appendix C

File: Appendix C - ACM query results.csv

Appendix D

File: Appendix D - IEEE query results.csv

Appendix E

File: Appendix E - SCOPUS query results.csv

Appendix F

File: Appendix F - SpringerLink Article query results.csv

Appendix G

File: Appendix G - SpringerLink Conference paper query results.csv

Appendix H

File: Appendix H - Web of Science query results.csv

Appendix I

File: Appendix I - SLR - Step 2 - duplicates removed.csv

Appendix J

File: Appendix J - SLR - Step 4 - Selected abstracts.csv

Appendix K

File: Appendix K - SLR - Step 5 - N 103.csv

Appendix L

File: Appendix L - SLR - Step 6 - SQ1 N 40.csv

Appendix M

File: Appendix M - SLR - SQ1 - RAW data.xlsx

Appendix N

File: Appendix N - SLR - SQ2 - Problems etc. N 14.csv

Appendix O

File: Appendix O - SLR - SQ2 - Contain CSF N 8.csv

Appendix P

File: Appendix P - Prerequisites CSF's cross-check.xlsx

Appendix Q

File: Appendix Q - Netherlands Code of Conduct for Research Integrity 2018.pdf

Appendix R

File: Appendix R - Declaration own work.pdf

Appendix S

Interview-protocol

Voorafgaand aan het interview:

1. Contact zoeken met contactpersoon/poortwachter van geselecteerde organisatie via email (delen van kennismakings pitch met contactpersoon/poortwachter zie (appendix V)), of hij/zij openstaat voor een kennismaking / interesse heeft in dit onderzoek.
2. Vragen om een afspraak te maken met de poortwachter om een pre-interview met hem/haar af te nemen om informatie te verzamelen betreffende de DevOps context en

samen aftasten of de organisatie voldoet aan de opgestelde selectiecriteria en of de poortwachter bereid is medewerkers aan te dragen voor interviews.
3. Sturen van toestemmingsformulier (appendix W) per email (appendix X) aan poortwachter.
4. Pre-interview met poortwachter

Voorafgaande aan het interview wordt het getekende toestemmingsformulier ingenomen en gevraagd of het goed is als dit interview wordt opgenomen.

Pre-interview guide met poortwachter

#	Vraag	Gerelateerd aan	Doel
1.	Kunt u aangeven wat de organisatie precies als IT-diensten/producten aanbiedt? Wordt de software van deze IT-diensten/producten door de organisatie zelf tot stand gebracht?	Organisatie - Selectiecriteria 1	Checken of de organisatie IT-diensten/producten levert (IT Service Provider) en kan worden vastgesteld of er aan software-voortbrenging wordt gedaan.
2.	Hoeveel medewerkers telt de organisatie (intern+extern)?	Organisatie - Selectiecriteria 2	Checken of de organisatie voldoet aan de gestelde criteria m.b.t. de organisatie grootte.
3.	Kunt u aangeven wat er binnen uw organisatie wordt verstaan onder DevOps? Zijn er DevOps teams aangesteld? Hoeveel? Kunt u aangeven hoe de rolverdeling nu is binnen het DevOps team? (grootte, verschillende rollen, expertise per rol, verantwoordelijkheden)	Organisatie - Selectiecriteria 3	Checken of de beschrijving in lijn is met de definitie van DevOps (Erich, 2019), of er DevOps teams zijn aangesteld en informatie ophalen m.b.t. de context van een DevOps team zoals (grootte, verschillende rollen, expertise per rol, verantwoordelijkheden)
4.	Kunt u aangeven hoelang geleden de implementatie van Continuous Delivery en/of Continuous Deployment succesvol was afgerond?	Organisatie - Selectiecriteria 4	Checken of de organisatie voldoet aan de gestelde criteria m.b.t. de minimale tijd tussen de afgeronde implementatie van Continuous Delivery en/of Continuous Deployment en dit onderzoek.
5.	Kunt u aangeven uit welke componenten de delivery pipeline bestaat binnen het DevOps team? (voorbeeld: build, integration, test, deploy, release) m.a.w. wat verstaat u onder een delivery pipeline? Kunt u aangeven welke rollen (expertise), op welke plaats in de delivery pipeline zijn gepositioneerd binnen het DevOps team?	Organisatie - Selectiecriteria 5	Informatie ophalen m.b.t. het beeld dat de organisatie heeft van een delivery pipeline, zodat er een duidelijk beeld geschetst kan worden van de rollen en op welk component in de delivery pipeline deze betrekking hebben. Dit moet bijdragen aan het reduceren van interpretatieverschillen.
6.	Kunt u aangeven welk(e) shared goal(s) jullie gesteld hebben binnen het DevOps team?	Organisatie - Selectiecriteria 6	Informatie ophalen m.b.t. de shared goal(s) die de organisatie / het DevOps team gesteld heeft.

7.	Kunt u aangeven welke performance eisen jullie hebben gesteld m.b.t. de/het gestelde shared goal(s)? Hoe meten jullie of jullie die performance eisen halen?	Organisatie - Selectiecriteria 7	Informatie ophalen m.b.t. de performance eisen aangaande de/het gestelde shared goal(s)? Checken of dit ergens wordt gemonitord of bijgehouden.
8.	Bent u bereid interview-kandidaten aan te dragen? Hoeveel zou u er aan kunnen dragen?	Organisatie - Selectiecriteria 8	Checken of er mogelijk genoeg interview-kandidaten beschikbaar zullen zijn.

Na het pre-interview met poortwachter:

5. Kennismakings afspraken maken met interviewkandidaten.
6. Tijdens de kennismakings afspraken zal ik mij kort voorstellen en uitleg geven over het afstudeeronderzoek en worden de interviewkandidaten informeel getoetst op de gestelde selectiecriteria voor een interview-kandidaat.

Voor de aanvang van pre-interviews met de interviewkandidaten zal gevraagd worden of het goed is als de interviews worden opgenomen.

Pre-interview guide met interview-kandidaten tijdens kennismakingsafspraak:

#	Vraag	Gerelateerd aan	Doel
1a.	Wat is je huidige rol binnen deze organisatie?	Respondent - Selectiecriteria 1	Validatie of de respondent deel uitmaakt van het DevOps team en matched met de rolverdeling gegeven door de poortwachter. Door te weten welke rol de geïnterviewde precies heeft, geeft dit meer inzicht in het perspectief van de geïnterviewde. Het kan bijvoorbeeld voorkomen dat er bias optreedt doordat de geïnterviewde bepaalde factoren die met zijn eigen werk te maken hebben belangrijker vind en deze misschien een hogere waardering geeft.
1b.	Hoeveel jaren werkervaring in deze rol heb je totaal?	Respondent - Selectiecriteria 2	Door te weten hoeveel jaren werkervaring in deze rol de geïnterviewde heeft, kan een indicatie geven over zijn kennis/ervaring (skill-niveau) m.b.t. zijn rol.
1c.	Hoeveel jaren werkervaring in deze rol heb je binnen deze organisatie?	Respondent - Selectiecriteria 3	Door te weten hoeveel jaren werkervaring in deze rol de geïnterviewde heeft binnen zijn huidige organisatie, kan een indicatie geven over zijn organisatie-kennis (gezien vanuit zijn rol).

1d.	Hoeveel jaren relevante werkervaring heb je m.b.t. DevOps en Continuous Delivery / Continuous Deployment?	Respondent - Selectiecriteria 4	Door te weten hoeveel jaren relevante werkervaring de geïnterviewde heeft met DevOps en Continuous Delivery / Continuous Deployment, kan een indicatie geven over zijn kennis/ervaring (skill-niveau) m.b.t. DevOps en Continuous Delivery / Continuous Deployment.
1e.	Hoeveel jaren relevante werkervaring heb je m.b.t. DevOps en Continuous Delivery / Continuous Deployment binnen deze organisatie?	Respondent - Selectiecriteria 5	Door te weten hoeveel jaren relevante werkervaring de geïnterviewde heeft met DevOps en Continuous Delivery / Continuous Deployment binnen deze organisatie, kan een indicatie geven over zijn betrokkenheid. (iemand kan bijvoorbeeld al 5 jaar bij de organisatie werken, maar sinds vorige maand in een DevOps team terecht is gekomen.)
1f.	Wat vind je van de DevOps en Continuous Practices implementatie in jullie organisatie? Vind je het succesvol? Of vind je het niet belangrijk? Of had de organisatie beter af kunnen zijn zonder al deze veranderingen?	Respondent - Selectiecriteria 6	Door te weten of de geïnterviewde een voorstander is van DevOps en Continuous Practices en deze als succesvol ervaart, straalt een optimistische houding uit. In het geval de geïnterviewde geen voorstander is van DevOps en Continuous Practices, kan er een indicatie zijn dat Continuous Delivery en/of Continuous Deployment wellicht niet goed of volledig geïmplementeerd is, de casus organisatie is dan niet geschikt.
1g.	Wil je deelnemen aan mijn onderzoek? Zou ik je dan mogen interviewen? Kunnen we dan een vervolgafspraak inplannen?	Respondent - Selectiecriteria 7	Bij genoeg interviewkandidaten (respondenten) die toestemming geven kan de organisatie geselecteerd worden als casus organisatie.

In geval voldoende interviewkandidaten aangeven bereid te zijn mee te willen werken aan de interviews en voldoen aan de gestelde criteria voor een interview-kandidaat:

Na de kennismakingsafspraak met interview-kandidaat:

7. Met de interviewkandidaten die voldoen aan deze gestelde criteria en toezeggen mee te willen werken, een nieuwe afspraak in te plannen met een interview-datum. Daarnaast wordt met een van de interviewkandidaten een afspraak gemaakt voor een pilot-interview.
8. Sturen van toestemmingsformulier (appendix W) per email (appendix X) aan geselecteerde interview-kandidaten inclusief voorbereidings slides met lijst potentiële kritische succesfactoren. (appendix U)

Pilot interview

9. Pilot interview wordt gehouden met een van de geselecteerde interview-kandidaten.
10. De interviewguide wordt eventueel nog aangepast aan de hand van de evaluatie van het pilot interview.
11. De interviews worden gehouden met de geselecteerde respondenten

Start interview

De duur van het interview wordt geschat op 60 minuten, om ervoor te zorgen dat er voldoende tijd is, zullen de interviews ingepland worden voor 90 minuten. De lijst met potentiële kritische succesfactoren zal worden uitgeprint, zodat deze gebruikt kan worden tijdens het interview ter ondersteuning. Appendix T wordt gebruikt door de interviewer ter ondersteuning bij vraag 2b, dat een lijst bevat met voorbeelden per kritische succesfactor in geval de respondent niet op een voorbeeld kan komen.

Voorafgaande aan het interview wordt het getekende toestemmingsformulier ingenomen en gevraagd of het goed is als dit interview wordt opgenomen.

Als eerste worden er twee vragen gesteld om informatie over de context van het DevOps-team op te halen:

2a.	Kunt u aangeven wat uw rol precies inhoudt? Zit daar ook een soort T-profiel aan?	Organisatie - Selectiecriteria 5	Informatie ophalen omtrent rol / T-profiel voor contextinformatie.
2b.	Kunt u aangeven wat het doel is van het DevOps team? (shared goal) M.a.w. waarvoor is het DevOps team opgericht? (bestaansrecht) Bijv. Een DevOps-team kan als doel hebben, “Het eenvoudig maken van test-automatisering, zodat andere teams hier makkelijk mee aan de slag kunnen, zonder al te veel specifieke kennis”.	Organisatie - Selectiecriteria 6	Informatie ophalen m.b.t. de shared goal(s) die de organisatie / het DevOps team gesteld heeft.
2c.	Kunt u aangeven welke performance eisen jullie hebben gesteld m.b.t. de/het gestelde shared goal(s)? Hoe meten jullie of jullie die performance eisen halen?	Organisatie - Selectiecriteria 7	Informatie ophalen m.b.t. de performance eisen aangaande de/het gestelde shared goal(s)? Checken of dit ergens wordt gemonitord of bijgehouden.

Per respondent zal het startpunt van onderstaande vragen verschillen. Dit wordt gedaan omdat er getracht wordt zoveel mogelijk potentiële CSF´s aan bod te laten komen en er hiermee voorkomen wordt dat door de gestelde tijdslimiet van het interview de potentiële CSF´s onderaan de lijst niet meer aan bod zouden komen. Het startpunt zal per respondent nader bepaald worden aan de hand van zijn expertise gebied. Bijv. bij een rol als “Architect” is het startpunt “Architecture”. * Vraag 2e en 2f zijn verdiepingsvragen en zullen in geval van tijdgebrek worden overgeslagen.

Start vragen m.b.t. lijst potentiële kritische succesfactoren (CSF's) zie (Appendix U) ---> (*deze factor*)			
2d.	Begrijp je de definitie van (*deze factor*)?	Validiteit	Door te weten of de geïnterviewde alle potentiële CSF´s begrijpt, kan een indicatie geven over de kwaliteit van de antwoorden. Als blijkt dat er nog potentiële CSF´s onduidelijk zijn, zullen deze opgehelderd worden.
2e.	Heb je (*deze factor*) ooit meegemaakt? En kun je een voorbeeld noemen? Zijn er misschien nog aanvullende factoren die hier een rol in hebben gespeeld?	SQ3/SQ2 (criteria 1 van een CSF)	Door te weten of de geïnterviewde de potentiële CSF mee heeft gemaakt en/of een voorbeeld kan noemen waaruit dit blijkt, kan criteria 1 van een CSF ("verified significant impact on the success of a continuous practice") worden bevestigd.
2f.	Kun je op een schaal van 1-5 aangeven wat de mate van belang is van (*deze factor*)? (1)Onbelangrijk (2)Enigszins belangrijk (3)Redelijk belangrijk (4)Belangrijk (5)Zeer belangrijk	SQ3	Door te weten of de geïnterviewde de potentiële CSF belangrijk vindt, kan een indicatie geven over het gewicht van de factor.
2g.	Waarom vind je (*deze factor*) van belang?	SQ3 (criteria 2 van een CSF)	Door te weten waarom de geïnterviewde de potentiële CSF van belang vindt, kan criteria 2 van een CSF ("results in verified performance improvements") worden bevestigd.
2h*.	Zou je je inschatting aangaande (*deze factor*) kunnen onderbouwen met (periodieke/ near real-time) metingen/rapportages?	SQ3 (criteria 3 van een CSF)	Door te weten hoe de organisatie de potentiële CSF hebben gemeten, kan criteria 3 van een CSF ("Success and performance may be measured") worden bevestigd.
2i*.	Kun je in het kort beschrijven hoe (*deze factor*) is aangepakt / geïmplementeerd?	SQ3	Deze vraag heeft als doel extra informatie op te halen hoe de factor is aangepakt / geïmplementeerd om meer inzicht te krijgen in de DevOps context.
Einde vragen m.b.t. lijst potentiële kritische succesfactoren (CSF's) zie (Appendix U)			

Einde interview

Na het interview:

12. De transcriptie wordt opgestuurd aan de respondenten die mee hebben gewerkt aan het interview ter bevestiging en controle of de transcripties juist zijn getranscribeerd.

Appendix T

File: Appendix T - Potential CSFs with examples.pdf

Appendix U

Lijst met potentiële kritische succesfactoren (CSF's) - Continuous Integration - / Delivery and Deployment	
Potential CSF	**Description**
Preconditions	Establishing the optimal provision of value (e.g. generating new capabilities, supporting routines and competencies, restructuring) for realization in use and context where standardization and routinization do not currently exist (Smyth, 2018). In other words affairs which are not under direct control of the continuous integration and -delivery/deployment process.
Goals	Clear goals for the teams migrating towards continuous practices and assimilation metrics.
Strategy and approach	Approaches to drive Continuous Integration and Continuous Delivery/-Deployment assimilation, and branching strategies.
Architecture	Diverse aspects on architecture of the product and related infrastructure.
Process design	Institutionalizing the process of continuous integration and -delivery/deployment and aspects of the process e.g., design, effort to initially setting up the proces, management, planning, sufficient time/resources, waste in the process and accuracy of the process.
Motivation	Motivation to adopt Continuous Integration, -Delivery/-Deployment and to get past early difficulties and effort, and discipline to commit often, test diligently, monitor the build status and fix problems as a team.
Resistance to change	Difficulty to change established organizational policies and cultures.
Complexity across customer organization boundary	No access to or control on a production environment or diversity and complexity of customer sites, which make it harder to fully automate the deployment process.

Acceptance by customer	Adopting the practice of continuous releases. Customer perception of their involvement in development and customer behaviour. Domain constraints. Feature discovery.
Sales and intermediaries	When user data is not accessible due to intermediaries.
Quality	Preserving quality and adequate documentation.
Customer involvement	Preparing and receiving customer input, establishing a customer sample group, and delivering feature growth.
Test complexity & source code control	Aspects on automating, configuring and using test environments, and source code control. Due to higher demands as compared to traditional (not agile) ways of work.
Coordination	Increased need for coordination between team members and multiple teams. Organizational structure. Co-locate by feature not discipline.
Communication	Intra and inter team communication, the right communication tools, awareness and transparency.
Knowledge and training	Sufficient proficiency, knowledge, skills and experience.
Tooling	Maturity of the tools and their surrounding infrastructure that sufficiently support the Continuous Integration, -Delivery/-Deployment process, including security, access controls and consensus among users about the choice of tools. Heterogeneous programming languages, operating systems and communication tools.
Pace	Improving the speed of delivering deployments to the customer.
Pressure	Increased pressure on the team to have code ready to be deployed immediately, and too much transparency which causes customers to interfere with developers' work.

Appendix V

Beste [Contactpersoon],

Ik ben op zoek naar een organisatie die bereid is mij te helpen met mijn afstudeeronderzoek.

De organisatie die ik zoek, heeft Continuous Integration - / Delivery and Deployment al geadopteerd en geïmplementeerd en wordt gewerkt met (een) DevOps team(s).

In mijn afstudeeronderzoek probeer ik antwoord te geven op de onderzoeksvraag: “Welke kritische succesfactoren zijn van toepassing op de adoptie en implementatie van Continuous Integration - / Delivery and Deployment in een DevOps context?”

De praktische relevantie is de teamleden bewust maken van de factoren die het succes van Continuous Integration - / Delivery and Deployment in hun organisatie kunnen beïnvloeden.

Het doel van dit onderzoek is het valideren van een lijst met kritische succesfactoren die ik heb samengesteld aan de hand van literatuuronderzoek.

Ik wil dit valideren door interviews te houden met DevOps teamleden die een nauwe betrokkenheid of rol hebben vervuld bij de implementatie van Continuous Integration - / Delivery and Deployment en wellicht betrokken managers.

Zou ik misschien langs mogen komen voor een oriënterend gesprek?

Met vriendelijke groet,

Robert Vonk

Appendix W

Toestemmingsverklaring

Inleiding
Geachte heer/mevrouw,
Wij vragen u om mee te doen aan een wetenschappelijk onderzoek. Meedoen is vrijwillig. Om u mee te laten doen, hebben wij wel uw schriftelijke toestemming nodig.

Voordat u beslist of u wilt meedoen aan dit onderzoek, krijgt u uitleg over wat het onderzoek inhoudt.
Lees deze informatie rustig door en vraag de onderzoeker uitleg als u vragen heeft. U kunt ook de hoofdonderzoeker, die aan het eind van deze toestemmingsverklaring genoemd wordt, om aanvullende informatie vragen.

1. Doel van het onderzoek
Zoveel mogelijk informatie ophalen omtrent Continuous Integration - / Delivery and Deployment in een DevOps context en kijken of de potentiële kritische succesfactoren herkent worden en gevalideerd kunnen worden.

2. Achtergrond van het onderzoek
Er is een theoretisch raamwerk opgesteld aan de hand van literatuuronderzoek met potentiële kritische succesfactoren die betrekking hebben op Continuous Integration - / Delivery and Deployment in een DevOps context. Deze lijst met potentiële kritische succesfactoren kunnen

vervolgens worden gebruikt als stuurmiddel om te zorgen dat Continuous Integration - / Delivery and Deployment succesvol geadopteerd en geïmplementeerd kan worden binnen een organisatie.

3. Wat meedoen inhoudt en wat wordt er van u verwacht
In dit onderzoek wordt er een interview bij u afgenomen. In dit interview probeert de onderzoeker zoveel mogelijk informatie omtrent de lijst met potentiële kritische succesfactoren op te halen. U dient antwoord te geven op de vragen die gesteld worden tijdens het interview zover dit voor u mogelijk is. Van u wordt verwacht dat u de complete lijst met potentiële kritische succesfactoren voor aanvang van het interview grondig bestudeert heeft, zodat eventuele onduidelijkheden vooraf opgehelderd kunnen worden. Dit interview zal worden opgenomen in de vorm van een audio-opname. Deze audio-opname zal worden getranscribeerd en de transcriptie zal vervolgens aan u worden toegestuurd per email ter verificatie. Van u wordt verwacht dat u de transcriptie controleert en goedkeurt indien deze juist is getranscribeerd. Deze goedkeuring kunt u per email geven.

4. Als u niet wilt meedoen of wilt stoppen met het onderzoek
U beslist zelf of u meedoet aan het onderzoek. Deelname is vrijwillig. Als u niet wilt deelnemen heeft dat geen nadelige gevolgen voor u. Als u wel meedoet, kunt u zich altijd bedenken en toch stoppen, ook tijdens het onderzoek. U hoeft niet te zeggen waarom u stopt. De gegevens die tot dat moment zijn verzameld, mogen worden gebruikt voor het onderzoek.

5. Einde van het onderzoek
Uw deelname aan het onderzoek stopt als het hele onderzoek is afgelopen als alle deelnemers klaar zijn.

6. Anonimiteit van uw gegevens
Voor dit onderzoek worden er geen persoonsgegevens verzameld. Alle participanten krijgen een code, zodat volledige anonimiteit wordt gegarandeerd.

7. Geen vergoeding voor meedoen
Het meedoen aan dit onderzoek is geheel vrijwillig en er zal geen enkele vorm van beloning worden verstrekt.

8. Heeft u vragen?
Bij vragen kunt u contact opnemen met [de onderzoeker/het onderzoeksteam].

9. Ondertekening toestemmingsformulier
Wanneer u voldoende bedenktijd heeft gehad, wordt u gevraagd te beslissen over deelname aan dit onderzoek. Door uw schriftelijke toestemming geeft u aan dat u de informatie heeft begrepen en instemt met deelname aan het onderzoek. Zowel uzelf als de onderzoeker ontvangen een getekende versie van deze toestemmingsverklaring.

_____	_____
Naam participant	Naam onderzoeker

_____ _____

Handtekening participant Handtekening onderzoeker

Datum:

[Onderzoeker]:
Robert Vonk
XXXXXXX
XXXXXXX

[Hoofdonderzoeker]:
Michiel van Belzen MSc
XXXXXXX
XXXXXXX

Appendix X

Allereerst ontzettend bedankt dat u mee wilt werken met mijn afstudeeronderzoek.

Ik heb u geselecteerd als interview-kandidaat aan de hand van uw nauwe betrokkenheid of rol die u heeft vervuld bij de implementatie van Continuous Integration - / Delivery and Deployment en uw huidige rol binnen een DevOps team. Het interview zal ongeveer 60 minuten in beslag nemen en ik zal een agenda reservering van 90 minuten inplannen, zodat we voldoende tijd hebben om door de lijst met potentiële kritische succesfactoren te lopen. Ik zou dan ook willen vragen om deze lijst met potentiële kritische succesfactoren (zie bijlage email) a.u.b. grondig te bestuderen zodat ik van te voren eventuele vragen / onduidelijkheden kan ophelderen om de kwaliteit van de antwoorden te maximaliseren.

Voor de aanvang van het interview heb ik uw uitdrukkelijke toestemming nodig voor het gebruik van de gegevens die voortkomen uit het interview. U kunt uw toestemming geven door het bijgevoegde document te ondertekenen en te overhandigen vóór de aanvang van het interview.

Als u nog vragen / opmerkingen heeft, dan hoor ik deze graag.

Met vriendelijke groet,

Robert Vonk

Appendix Y

File: Appendix Y - Demographic information about the respondents.xlsx

Appendix Z

File: Appendix Z - CSF Data Synthesis v3.xlsx

Appendix ZA

CSF - Preconditions

Description of definition

Establishing the optimal provision of value (e.g. generating new capabilities, supporting routines and competencies, restructuring) for realization in use and context where standardization and routinization do not currently exist (Smyth, 2018). In other words affairs which are not under direct control of the continuous integration and -delivery/deployment process.

Clear definition

With the exception of one respondent, the definition was immediately clear for all respondents (5). The respondent who did not understand the definition immediately, did understand the definition after the interviewer clarified the definition. Due to lack of time, at 1 respondent the interviewer did not have enough time to present this CSF to him/her.

Experience

3 respondents indicate to have experienced this CSF. 2 respondents indicate that they did not experience this CSF, assuming that: "Alleen ik kan hier niet voorbeelden bij noemen waar wij iets mee hebben gedaan." because they can't present any examples.

Examples

Respondent 3 indicates as an example of a precondition that he is often dependent on other teams, he is often dependent on other teams, who have to make a certain service:

“Dat betekent dat er een stuk autorisatie automatisch moet worden geregeld en daarvoor moet een service gemaakt worden, die dat automatisch doorvoert en daarmee zijn we afhankelijk van het “portalen-team” om die service te maken.”

In addition, the respondent also describes a problem that occurs here, namely that it is not fast enough and takes a long time:

“dus dat komt dan op de backlog, maar vaak bijt dat, vaak gaat dat niet snel genoeg.” which means there's a good chance they won't make the sprint, because they have to wait for the other team:

“Hetzelfde hebben we met het ontsluiten van data naar ons zelf toe. Als het onze eigen database is, dan kunnen we daar direct bij. Zodra wij data samengevoegd uit meerdere databases nodig hebben. Dan moet dat via een centrale of een gegevensmagazijn wordt dat hier genoemd. En dat is een projectgroep die dan een service maakt, die wij kunnen aanroepen voor die gegevens. Ook daar zien

we vaak, van "nou we willen het toch een beetje anders, we willen een wijziging hebben" dat soms heel lang duurt. Waardoor wij de sprint niet halen."

Respondent 6 indicates that the use of "Liquibase" is a precondition for Continuous Delivery. "Liquibase" is used to execute automated database mutations:

"Het voorbeeld is gewoon eigenlijk het project waar we nu mee bezig zijn en dat is "ABL", en dat werd eerst niet Continuous gedelivered, omdat we bijvoorbeeld niet gebruik maakten van Liquibase, he dat is een tool, een script waarmee je automatisch database mutaties kan uitvoeren. Daar moest eerst allemaal met ingewikkelde toestanden via.., wij noemden het dan vrachtbrieven zeg maar, dat waren uitgebreide excel-sheets en dat moest dan door een bepaalde afdeling moest dan de wijziging doorgevoerd worden. Dat was allemaal moeilijk, moeilijk, moeilijk. Tegenwoordig kan dat met Liquibase en dat is al een onderdeel van je deployment. En dat gebruiken we natuurlijk in ABL-nieuw wel. Maar vroeger konden we niet database changes continuous deliveren."

Respondent 4 indicates as an example of a precondition that the "use case descriptions" and "UX-designs" need to be defined, because they are not always clear and often change:

"De use case beschrijving onder andere,... die is niet altijd even helder, maar ook UX designs die wel eens wisselen... Ja, die gaan nog wel eens een keer, dat wisselt nog wel eens" "dat zijn echt de scherm ontwerpen. Hoe ze willen dat het gaat worden / eruit gaat zien. Dat is vaak onduidelijk. En dat komt dus ook wel doordat.. In dit specifieke project de eisen vanuit Den Haag nouja, best wel vaak weer wijzigen. Dus daar ook niet altijd helder hadden, van wat ze nu precies willen." "Voor de duur van de Sprint zal eigenlijk zo'n ontwerp vast moeten staan."

Weights

Table 9.1 - Preconditions - weights

	Respondent 1	Respondent 2	Respondent 3	Respondent 4	Respondent 5	Respondent 6
Weights	-	-	5	3	-	5

The reason why respondent 3 thinks this CSF scores a 5 and is therefore seen as very important is because he/she indicates that it is his/her concern, towards the user. He/she considers him/herself responsible for the result, he/she has sent to that user. Therefore he/she think it is inappropriate that the user has to wait very long: "*Dat dat een weekje of 2 blijft hangen, en dat wil je niet he want ik ben daar wel verantwoordelijk voor het resultaat wat ik gestuurd heb naar die gebruiker.*", because he/she depends on others (different team(s)). He/she also indicates that this user will end up with him/her after all: "*Ja, dat vind ik toch wel heel belangrijk" want die gebruiker komt bij mij uit, uiteindelijk.*"

The reason why respondent 6 thinks this CSF scores a 5 and is therefore seen as very important is because he/she indicates that this CSF can be seen as a precondition, just like the name of the CSF: *"Ja, omdat het een precondition is…de term zegt het zelf al."*

The reason why respondent 4 thinks this CSF scores a 3 and is therefore seen as relatively important is because he/she indicates that by defining the design for a sprint, they probably can achieve the sprint: *"Afstemming maar ook dat je gewoon gedurende het traject dat je ermee bezig bent, je meters kan maken om die Sprint weer te halen. En dat het niet zo is dat je, iedere keer weer terug moet vallen op wat anders. Voor de duur van de Sprint zal eigenlijk zo'n ontwerp vast moeten staan."*

Performance improvements

No performance improvements were observed.

Measurement of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the measurement of the CSF.

Implementation of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the implementation of this CSF.

Contextual relationships

It is observed that the example of the precondition given by respondent 4 to "define the design" might have a relationship with the performance requirement to "achieve the sprint".

Conclusion

This potential CSF could not be seen as verified CSF, because the CSF meets only criteria 1.

CSF - Goals

Description of definition

Clear goals for the teams migrating towards continuous practices and assimilation metrics.

Clear definition

For 1 respondent the definition was immediately clear, for the other respondents (3) it is unclear whether this definition is clear. This was not explicitly asked. However, it can be deduced that the definition was clear, because they could immediately indicate if they had experienced this. Due to lack of time, at 2 respondents the interviewer did not have enough time to present this CSF to him/her.

Experience

2 respondents indicate to have experienced this CSF. 2 other respondents indicate that they do not have experienced this CSF.

Examples

Respondent 6 gives an example of a goal to set up a Jenkins pipeline at all. So to set up the tooling in order to make Continuous Delivery a reality and to make it possible. According to the respondent, this has been a goal from the beginning.

Respondent 2 indicates that they were guided by the company Xebia Labs at the beginning of their Continuous Delivery journey using the deploy tool: "XL Deploy" which contains "Continuous Delivery Maturity Metrics", according to the respondent, goals could result from this if it appears that certain aspects of Continuous Delivery score low.

The respondent also describes how "Continuous Delivery Maturity Metrics" works:

"Daarin staan een soort pilaren die gaan over het Build, CI gedeelte, test gedeelte en cultuur is een kolom, zo zijn er aantal aspecten die zeg maar bekeken worden en daarvan hebben ze de maturity levels zeg maar beschreven. Onderin de kolom staat een tekst dat als je nog helemaal niks aan build automation doet en als je volledig geautomatiseerd doet, dat staat bovenin de kolom van het build aspect. En wat je daarmee kan doen is heel simpel zichtbaar maken op welk gebied je nog extra effort moet, bij ons was bijvoorbeeld daarmee heel snel zichtbaar dat we met test automation ver achterliepen. Terwijl dat natuurlijk gewoon een heel belangrijk onderdeel is."

After that, the respondent indicates that they have used the "Continuous Delivery Maturity Metrics" to clarify the management where to put effort and resources:

"Waar we het voor gebruikt hebben, is om het management duidelijk te maken waar zeg maar extra effort en resources opgezet moeten worden. Dus als je bij het management komt met een verzoek voor een doel, dan willen ze weten waarom. Nou wij kunnen dat aantonen op basis van zo'n Maturity matrix, dat we een test tool nodig hebben omdat te automatiseren bijvoorbeeld. En experts binnenhalen om ons daarbij te helpen."

Weights

Table 9.2 - Goals - weights

	Respondent 1	Respondent 2	Respondent 3	Respondent 4	Respondent 5	Respondent 6
Weights	-	3	-	1	4.5	4

The reason why respondent 2 thinks this CSF scores a 3 and is therefore seen as relatively important is because he/she indicates that it results in focus. He/she indicates that it could be used as an instrument to convince the management of where the focus should be at the moment: *"Als je aan een reis begint, dan heb je nog zoveel op je verlanglijstje staan, van wat je wilt aan goals, van wat je wilt halen. De kans is groot dat je niet alles in 1 keer gaat halen, en wij hebben gewoon een lange reis, ik bedoel wij zijn niks voor niks overheid, een beetje gekscherend bedoeld in de zin van dat gaat allemaal niet super snel. Er zit niet een commerciële drive achter en wij hebben ook te maken met wetgevingen en andere zaken zeg maar die een rol spelen die ook om prioriteit vragen. En je moet dus met een goed verhaal komen, om iets voor elkaar te krijgen, en je krijgt niet voor elkaar om een heel jaar lang met een complete 600 man ICT afdeling, zich helemaal alleen maar te focussen op van: "We gaan nou even dit jaar Continuous Delivery doen. " nee, daar zijn we dus sinds 2000 eigenlijk mee bezig. En heel langzaam zetten we stapje voor stapje, en we zijn nog niet aan het eind. Maar met die goals, dat is het instrument wat je kan gebruiken om het management te overtuigen van waar op dit moment de focus zou moeten liggen.*"

The reason why respondent 4 thinks this CSF scores a 1 and is therefore seen as unimportant is because he/she indicates that he/she loves the "doing" approach: *"Nouja, ik hou wel een beetje van de aanpak van: "Laten we het gewoon proberen! en gewoon doen!" En dan zien we wel, waar we komen! En waar het niet werkt, kun je altijd weer wat anders doen."*

Performance improvements

No performance improvements were observed.

Measurement of the CSF
Respondent 6 indicates that there should be a "closed user story" which indicates that the goal to design the "Jenkins pipeline" is achieved:

"Nou, niet anders dan dat er destijds ook een user story voor geweest zou zijn, om die pipeline in te richten, en die user story heeft natuurlijk ook gewoon een hele flow doorlopen met "Wanneer die is opgepakt door een ontwikkelaar" en "Wanneer die is getest" en "is gereed gemeld" zeg maar, dus dat is het dan ook. Geen uitgebreide rapportages ofzo. Je kan hooguit zeggen van: "Die story is toen opgepakt en toen gereed gekomen"."

Respondent 2 indicates that the "Continuous Delivery Maturity Matrix" is just a .pdf document, where you have to draw a line when the description of the maturity level is equal to the reality: *"Nou die tool is een papiertje hoor, als het hebt over die Maturity matrix. Dat is een pdf die je download, en waar je gewoon letterlijk de streep zet, als daar een omschrijving staat die past bij jouw niveau op dat moment, is dat het niveau wat je op dat moment hebt"*

Implementation of the CSF
Due to a lack of time, the interviewer did not have enough time to ask about the implementation of this CSF.

Conclusion

This potential CSF could not be seen as verified CSF, because the CSF meets only criteria 1. Criteria 3 could not be confirmed because no measurement report or document was provided to prove this.

CSF - Strategy and approach

Description of definition

Approaches to drive Continuous Integration and Continuous Delivery/-Deployment assimilation, and branching strategies.

Clear definition

4 respondents indicated that the definition was immediately clear. Due to lack of time, at 2 respondents the interviewer did not have enough time to present this CSF to him/her. Respondent 2

mentioned that he thinks the factor actually consists of 2 parts: "*kwa strategie en approach, dat zie ik wel als 2 verschillende dingen.*"

Experience

3 respondents indicate that they do not have experienced this CSF. Respondent 2 indicates: *"Iedereen kon doen wat die wilde vanaf het begin af aan. Dus er ligt geen strategie aan ten grondslag."* But he does indicate to have experienced "an approach".

Examples

Respondent 2 gives as an example that the case organization used a tech-driven approach, which can be seen as a bottom-up approach: *"Kijk, je kunt een approach hebben, zonder strategie. Ik kan je vertellen, dat is zoals (de case organisatie) het aangevlogen heeft. Mijn interpretatie van strategie is, dat er op management-niveau een strategische beslissing genomen wordt, van we willen dit bereiken, maar wij zijn op een moment begonnen dat het ook daadwerkelijk in de buitenwereld begon. Ik heb tech-driven erbij gezet.. Ik bedoel meer dat het door technische mensen, niet door de techniek, maar door de technische mensen zeg maar geïnitieerd is. Op de werkvloer werd het populair en interessant gevonden om daarmee bezig te gaan en het is langzamerhand zeg maar richting management., bij het management doorgedrongen zeg maar."* Respondent 5 also indicates to recognize the bottom-up approach.

Weights

Table 9.3 - Strategy and approach - weights

	Respondent 1	Respondent 2	Respondent 3	Respondent 4	Respondent 5	Respondent 6
Weights	-	4	-	4	1	-

The reason why respondent 2 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that a strategy is not necessary and a bottom-up approach is important because Continuous Integration and Continuous Delivery is a relatively technical story and often does not receive the necessary attention from management and is therefore not imposed from above in the form of a strategy: *"Als we het hebben over Continuous Integration, Continuous Delivery dan is een .. kwa natures zeg maar een behoorlijk technisch verhaal. Het gaat ook over automatisering en managers.. Ons management is niet vaak technisch opgeleid, maar het zijn echte managers en daar verwachten wij dat gewoon niet van. Dus wordt het niet van bovenaf zeg maar opgelegd, maar komt het van onder...Wat ik net vertelde was meer de waarom het een bottom-up was, maar de strategie zeg maar, zoals wij, wij hebben gewoon de vrijheid binnen (de case organisatie) om dat soort strategische keuzes ook te mogen maken."*

Performance improvements

No performance improvements were observed.

Measurement of the CSF

Respondent 2 indicates that technical architecture documents are being used for the approach:

"Schrijf ik daar technische architectuur stukken over. En er zijn stukken die ik op uit eigen beweging geschreven heb om dit in gang te zetten, om Continuous Delivery voor elkaar te kunnen krijgen."

Implementation of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the implementation of this CSF.

Conclusion

This potential CSF could not be seen as verified CSF, because the CSF meets only criteria 1. Criteria 3 could not be confirmed because no measurement report or document was provided to prove this.

CSF - Architecture

Description of definition

Diverse aspects on architecture of the product and related infrastructure.

Clear definition

All respondents (6) indicates that the definition was immediately clear. Respondent 2 indicate that he/she distinguish between the product and the infrastructure: "*Als ik het zeg maar moet plaatsen binnen een Continuous Delivery / Continuous Integration verhaal, vind ik het met name op infrastructuur, is mijn ervaring, dat dat een grote factor is. Op product-niveau eigenlijk niet. Wij zijn van mening dat Continuous Integration / Continuous Delivery het tool gebeuren daar, wat dat mogelijk maakt, het maakt niet uit welk product je daar in stopt, dat moet er maar mee om kunnen gaan. Je kunt een heel slechte productarchitectuur hebben en dat toch met Continuous Delivery en Continuous Integration behandelen.*"

Experience

All respondents indicate that they have experienced this CSF.

Examples

Respondent 1 indicates that they have a separate club of architects, who create the architecture. Six months ago they found out that the architecture is very well designed, but the infrastructure is not yet there at all. So their plans what they wanted are there, but it has not yet been realized, so there is a discrepancy between the designs and reality: "*Dus een club waarvan we verwachten dat die iets heeft wat we kunnen gebruiken hebben dat helemaal niet. Dus we hebben gewoon een gescheiden club die het uitdenkt en ons ook eigenlijk opleggen, van: "Oh nieuwe applicatie? Dan zou je het eigenlijk zo en zo moeten doen." en wij volgen he, wij zijn het braafste jongetje van de klas. En we zijn er ook blijkbaar achter gekomen dat we de enige braafste jongetje zijn. En het probleem daarin is is dat ze gewoon geen mandaat hebben, geen geld, maar op managementniveau wordt naar hen geluisterd. Dus er wordt gedacht dat het er is, maar het is er niet.*"

The above example given by respondent 1 is also confirmed by respondent 3: "*Een van de eerdere registers die ik gemaakt heb, waar werd gezegd van: "Ja, waarom moeten we dit allemaal doen?" En dan krijg je dus de uitleg van die architectuur die hebben we, om deze redenen. Maar dan moet die architectuur wel bestaan en niet te flexibel zijn.*"

The above example given by respondent 1 is also confirmed by respondent 6: "*Wij willen graag voldoen aan de XXXXXXX-architectuur, zoals die is geformuleerd. En nou daar doen we ook ons best voor, en we merken ook dat die af en toe nog niet helemaal volwassen is, in de zin van er mist wel eens een component. He bepaalde generieke componenten die missen we, of de registers zijn niet helemaal aangesloten op de manier zoals dat door de architectuur wordt voorgeschreven. En dat beperkt ons dan natuurlijk.*"

Weights

Table 9.4 - Architecture - weights

	Respondent 1	Respondent 2	Respondent 3	Respondent 4	Respondent 5	Respondent 6
Weights	4	3	4	4	4	2

The reason why respondent 5 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that if you stay within the established boundaries of the architecture, you can simply deliver faster. Because you don't have to discuss everywhere. According to the respondent, it takes a lot of time and lobbying to get things done that fall outside the boundaries. In that sense he thinks architecture is important:

"Als je gewoon de architecturale keuzes maakt, dan liggen dingen gewoon vast en daar is over nagedacht en ook vastgelegd. En dan weet je van te voren waar je rekening mee moet houden als je iets doet. Dus die kaders die zijn, soms zijn ze vervelend, omdat je in een kader moet. Maar zolang je in het kader loopt, heb je verder geen gedoe. Dat is de vastgelegde route om producten te maken en op te leveren. En als dat allemaal onduidelijk is, dan is er elke keer gedoe. "Ja ik ken het product" ja, dit zijn de kaders, daar blijf ik binnen. En dan heb je geen gedoe, en kun je gewoon sneller, je product ook opleveren.Nou duidelijkheid verschaffen in de kaders waarbinnen je moet opereren. Net als inderdaad, als doel, als je dat doet, en die kaders zijn breed genoeg om daar binnen nog fatsoenlijk kan opereren uiteraard, soms niet maar meestal wel. Dan kun je gewoon sneller opleveren. Want je hoeft niet overal te discussiëren. Nou, ik kan je wel vertellen, als je buiten de kaders valt, dan duurt het zolang allemaal, dus dit is echt belangrijk. Ja het release proces, door de hele ontwikkelstraat heen, tot en met productie. En als je buiten die kaders zit, dan moeten er nieuwe dingen worden bedacht, en ja jij wilt dat, maar daar hebben wij niet in voorzien, want dat valt buiten de kaders. Daar moet dan over nagedacht worden, en ook niet door 1 persoon, XXXXXXXX is nogal groot, en iedereen heeft over van alles wat te zeggen. Dan moet je bij een heleboel partijen bij langs, beveiliging, en weet ik veel allerlei mensen, die moet je allemaal met jouw plan, moet je ze meenemen. En daar een stempeltje op krijgen van: "Nou dat is goed dat je dat doet" en dat kost heel veel tijd, doorlooptijd met name! Lobbytijd! Lobbytijd en doorlooptijd! Dingen gaan, gaat niks van zelf hier, dus je moet daar gewoon voor lobbyen, duidelijk maken wat je wil, waarom je het wil, en gaan ze ook nog vragen: "Ja maar waarom wil je dat dan zo, waarom blijf je niet binnen de kaders?, Wat is jouw motivatie om

buiten die kaders te gaan? Die kaders zijn er toch niet voor niks? Daar is over nagedacht." nou dan moet je dat allemaal, dus dat kost lobbytijd en doorlooptijd, heel lang."

The reason why respondent 1 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that it contributes to develop things a lot faster: "*Theoretisch gezien, belangrijk, als het klopt wat ze hebben en je kunt er gebruik van maken zou je veel sneller kunnen ontwikkelen, onze tijd zit heel veel in de connectie met al die andere partijen. Ik denk dat als, de architectuur goed neergezet wordt maar ook daadwerkelijk er is, zoals beschreven is, Ik denk dat als het er goed staat, en het is voor iedereen duidelijk, je gewoon veel sneller nieuwe dingen kunt maken.*"

The reason why respondent 6 thinks this CSF scores a 2 and is therefore seen as slightly important is because he/she indicates that even without architecture you still can deliver continuously: "Ik denk dat je zonder die architectuur ook prima Continuous kan deliveren, misschien niet helemaal waar iedereen blij van wordt. Maar joh, kan prima. Als het gaat om Continuous Delivery."

Performance improvements

Both respondents indicate that this CSF will implies speed.

Respondent 1: "*Je gewoon veel sneller nieuwe dingen kunt maken (ontwikkelen).*"
Respondent 5: "*Dan kun je gewoon sneller opleveren.*"

Measurement of the CSF
Due to a lack of time, the interviewer did not have enough time to ask about the measurement of the CSF.

Implementation of the CSF
Due to a lack of time, the interviewer did not have enough time to ask about the implementation of this CSF.

Conclusion

This potential CSF could not be seen as verified CSF, because the CSF meets only criteria 1 and 2.

CSF - Process design

Description of definition

Institutionalizing the process of continuous integration and -delivery/deployment and aspects of the process e.g., design, effort to initially setting up the process, management, planning, sufficient time/resources, waste in the process and accuracy of the process.

Clear definition

For 4 respondents the definition was immediately clear. 1 respondent had doubts, but after the definition has been clarified, it was immediately clear. Due to lack of time, at 1 respondent the interviewer did not have enough time to present this CSF to him/her.

Experience

4 respondents indicate that they have experienced this CSF. 1 respondent indicates that he/she does not experienced this CSF.

Examples

Respondent 6 indicates that the process design is embedded in the tooling: *"Ja ik denk dat dit gewoon onze Jenkins pipelines zijn, in combinatie met de hele workflow in XL-Release, he de pipelines.. Interviewer: Dat wordt daarin geborgd zeg je eigenlijk? Geïnterviewde 1: Ja dat denk ik wel, de pipeline is meer voor het "builden" en..ja in ons geval ook deployen op een testomgeving. En XL-Release komt om de hoek kijken op het moment dat het verder de straat op moet he, dus op het moment dat een release 1 keer akkoord is, dan gaat die via XL-Release verder de straat doorgezet worden. En dan moeten ook mensen akkoord geven, en dat soort dingen meer, met vinkjes, en daar zit een hele workflow, zit eronder. En op die manier komt het uiteindelijk in productie te staan. Interviewer: Oke, dus eigenlijk, dus het process design, deze factor, zeg je eigenlijk die is geborgd in de tooling? Geïnterviewde 1: Ja."* This example is also confirmed by Respondent 4.

Respondent 5 indicates that they have a full support team working on the tooling around enabling Continuous Integration and Continuous Delivery "The Continuous Delivery team" and, according to the respondent, is institutionalized during departmental consultation meetings: *"Er is een heel team die die tooling beschikbaar stelt en ook de kennis daarvan heeft. En ik ben dus afnemer eigenlijk he en dat wordt dan geïnstitutionaliseerd omdat dat gewoon tijdens vakgroep overlegvergaderingen, alle programmeurs bij elkaar zitten, worden er gewoon presentaties over gegeven. En zodoende is het bij iedereen bekend ook. Als je er al niet zelf mee werkt."*

Respondent 3 indicates that there is a provisioning thing with which you can create a silo and where components can be added to it. From that silo you can, according to the respondent, perform tests, check in code in a repository, build, indicate which checks need to be done in the Jenkins pipeline and automatically transfer it to a production silo via certain tooling: *"Ik weet dat er een bepaald proces nu is, XXXXXX heb je net gesproken, die heeft een provisioning ding gemaakt, waarmee wij een zogenaamde silo mee kunnen maken, waar wij componenten aan toe kunnen voegen. En vanuit die silo, dat is een belangrijk punt, daar kunnen we dus in testen, code checken we in, in een repository, en vanuit die repository kunnen we via Jenkins gaan bouwen en in Jenkins vervolgens in een pipeline daarin ook aangeven van "We willen die en die controles doen" en er is tooling in place, als al die controles goed gaan, het ook automatisch door te zetten, naar 1 van die silo's. Waarmee je dus Continuous Delivery kunt doen, dat moet een productie silo zijn, he dan komt het dus gelijk operationeel. Dus dat proces dat is er, hoe je dat doet. Wij maken er ook gebruik van, wij zijn binnen het project nog niet zover, dat we ook automatisch op productie deployen."*

Respondent 2 indicates that they are developing a "pipeline template" by means of the "Jenkins template engine" which ensures that hack tests will be institutionalized: *"We hebben iets dat heet een verlicht hacktest regime. Bij XXXXXXX moet ieder product wat naar productie gaat een hack-test ondergaan. Gewoon een zware test omdat het echt, echt veilig is. En als jij bepaalde rapporten kunt over leveren die uit geautomatiseerde tooling komen, en vervolgens daaraan voldoet aan bepaalde*

normering van de resultaten daarvan. Dan kun je vrijstelling krijgen. Als jij dat een aantal keer zeg maar aantoont. Dat jij die tooling zeg maar netjes gebruikt. En dat ze kunnen zien wat de toestand is van die applicatie. Dan krijg je een verlichte regime. Dat betekent dat je niet voor iedere release naar productie, een hack-test nodig hebt. Waar ik nu op dit moment mee bezig ben is dus met een Jenkins template engine een pipeline template te maken. Waarin dat dus wel geïnstitutionaliseerd wordt."

Respondent 2 also indicates that by institutionalizing this you can start working with something called "Staging & Promotion" staging and promotion means that you can divide your process into phases. In which you can easily make a distinction, because everyone has implemented their quality gates in the same way, because it is institutionalized. This allows you to measure per stage which artifacts are still in which stage: *"Wij hebben voorheen toen we nog Continuous Integration deden, zonder Continuous Delivery hadden we 2 repositories, eentje met snapshots, dat zijn de probeer-artifacts, wat nog niet productierijp is. En als het rijp was, maak je er een release van. En daar hadden we ook een repository voor. Maar die laatste repository is eigenlijk de enige die nu nog gebruikt wordt, want we maken eigenlijk geen snapshots meer, dat is namelijk Continuous Delivery, alles wat je bouwt, is productierijp. Totdat er ergens een testje zegt van, "Dit klopt niet." Dan keur je hem af. Wat je krijgt, is dus die ene repository, daar werd alles ingepompt en dan met een tempo 10/20/30x hoger dan voorheen. En we gingen maar schijfruimte toevoegen, schijfruimte toevoegen, we zaten op een gegeven moment op 20 Terabyte, jaa niet te managen… en wij kunnen dus door dit te institutionaliseren kun je gaan werken met iets dat heet "Staging & promotion" staging en promotion betekent dat je jouw proces kunt opdelen in fase. Ja het staat bekend om de term: "staging en promotion", maar het punt is dat je artifacts verplaatst, en je hebt dus inzicht in wat in productie zit, dat zit namelijk in jouw productie-repository..moet je wel zorgen dat daar wel die schoning opzit, maar die schoning die we nu niet hebben zeg maar, waar het probleem door ontstaat, is dat doordat er geen schoning is. Nu kunnen we schonen, want we weten nu wat troep is. Dat bestaat niet als er 2 of 3 maanden, misschien 4 maanden en we moeten ergens een grens trekken.. Dan kun je zeggen van ja, dit niet.. dat 4 maanden stilgestaan heeft dat mag weg."*

Weights

Table 9.5 - Process design - weights

	Respondent 1	Respondent 2	Respondent 3	Respondent 4	Respondent 5	Respondent 6
Weights	-	4	4	4	5	4

The reason why respondent 6 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that having such a process can be seen as a prerequisite: *"Het is randvoorwaardelijk! Ja of je moet het allemaal met de hand gaan doen, dat zou ook nog kunnen, maarja dat wordt dan een beetje veel, omdat dan continu te doen natuurlijk."*

The reason why respondent 5 thinks this CSF scores a 5 and is therefore seen as very important is because he/she indicates that it is very important to have a process, because he/she does not have the expertise to do it on their own. *"Nou als ik dat niet heb, dan lukt het mij niet eens. Ik heb zelf niet de expertise om, als ik hun tooling niet had, en er wordt tegen mij gezegd, nou "Doe even een Continuous Delivery.." nou dat gaat niet, want ik weet niet hoe dat moet. En ik heb ook geen zin, om me erin te verdiepen. He, je kunt niet alles he. Ik ben meer van, ik ben meer een bouwer, die gewoon*

functionele wensen van de klant om wil zetten in een werkbaar product, en dat je daar tooling voor nodig hebt, nou dat is allemaal prima. Maar, geef mij het maar."

The reason why respondent 4 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that having such a process is necessary because you don't know what to do, and what the right procedure is: "*Omdat je anders niet weet wat je moet doen. En hoe die stappen elkaar op moeten volgen."*

The reason why respondent 3 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that the process design makes sure that you don't step into certain pitfalls, which you can easily skip without that tooling. Things can slip through. For example security aspects. The process design that forces you to follow the process. The respondent gives an example: *"Op het moment dat je doorgaat naar een volgende omgeving, dus een definitieve build maakt van development en beschikbaar komt op test, mogen die testen gewoon niet meer falen. Terwijl als je dat allemaal achterwege laat, dan blijkt dat er een unit-test faalt, "ja boeiend zeg je dan, dat is een unit-test" maar die faalt omdat er in productiecode iets mis is. En dan kan het er dus, als je zegt "Nou, deploy hem maar gelijk op dat systeem vanuit zo'n ontwikkel plek!" dan omzeil je dus dat hele spul, dan kan je dus heel makkelijk bugs op productie introduceren. En dat maakt het voor mij wel heel belangrijk om daartegen te beschermen."*

The reason why respondent 2 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that the freedom you have, if you don't institutionalize it, leads to abuse. The respondent gives an example: *"Als je pijn hebt van SonarQube als zijnde geautomatiseerde stap aan het begin van je proces en je plakt hem dan maar gewoon aan het eind. Dan kun je de rest van die stappen ook uitvoeren. Ja dat is niet zoals het zou moeten zijn. Je zou daar gewoon over na moeten denken, en ik merk ook bij mensen, dat als je het erover hebt. Van wat is nu een logische volgorde waarin zaken plaatsvinden? Dat bij iedereen het wel een beetje op hetzelfde uitkomt. Dus de mogelijkheid om het te institutionaliseren die is best wel aanwezig. Het is gewoon niet gedaan en dat is gewoon een kwestie van daar effort insteken."*

Performance improvements

No performance improvements were observed.

Measurement of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the measurement of the CSF.

Implementation of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the implementation of this CSF.

Conclusion

This potential CSF could not be seen as verified CSF, because the CSF meets only criteria 1.

CSF - Motivation

Description of definition

Motivation to adopt Continuous Integration, -Delivery/-Deployment and to get past early difficulties and effort, and discipline to commit often, test diligently, monitor the build status and fix problems as a team.

Clear definition

4 respondents indicate that the definition was immediately clear. 1 respondent did not understand the definition, because he/she does not understand this specifically in relation to Continuous Delivery. He/she thinks it's more obvious. "*Maar ik vind dit wel een beetje een open deur, want dat geldt natuurlijk altijd. Niet alleen voor Continuous Delivery, maar alles, bij alles wat je doet toch*" Due to lack of time, at 1 respondent the interviewer did not have enough time to present this CSF to him/her.

Experience

All respondents (5) indicate that they have experienced this CSF.

Examples

Respondent 6 indicates that as a team they fix problems and are highly motivated, they have a monitor that shows the status of the build which is being monitored: *"Ja wij zijn als team zeer gemotiveerd, wij hebben nu ook een monitor, een algemene monitor staan, waarop de status van de build te zien is. Zodra dat ding niet meer groen is, dan gaan we dat fixen. Naja in het verleden hadden we ook, als iemand de build 3x stuk gemaakt heeft moet die trakteren dat soort grapjes zeg maar. Dus dat zijn er ook wel voorbeelden van. Over het algemeen als team zijn we gemotiveerd, om gewoon dat in orde te houden."*

Respondent 4 also indicates that his team monitor the build status properly: *"Kwa monitoren van de build status ja, dat is bij ons allemaal keurig ingericht. Met een Jenkins pipeline, die gaat gewoon automatisch door totdat die op een punt komt van: "Hee dit is niet goed.. Ik ga niet verder" nou dan moet je dan eerst fixen, anders krijg je daar geen release-build van."*

Respondent 3 indicates that they strive to check-in their work at least once a day, regularly runs tests and if you don't know something, asks someone else for help: *"Het is een afspraak onderling. We noemen het meestal de "onder de tram reden" he, want dan ben je ineens een paar dagen eruit en het kan niet zo zijn dat er dan niemand verder kan met jouw werk. Dus het moet centraal ergens staan. Dat betekent ook dat je regelmatig je unit-testen draait, en dat zijn de kleine testen, zodra we iets opleveren voor Test, worden ook integratie-testen enzo ook gedaan. En binnen een sprint zie je toch dat we 3 a 4 versies maken van iets. Die we aanbieden aan Test. Dat al die testen ook gedraaid worden. En dit zijn er dan 3 a 4, omdat het weer terugkomt, met van, "Dit gaat niet goed, of dat gaan niet goed" , zegt niks over de kwaliteit van het ontwikkelen, maarja het zijn dingen die kunnen voorkomen. En als er problemen zijn, waar je niet uitkomt, dan. Dat is eigenlijk een standaard Scrum practice, of een Agile practice, van als je het niet weet, vraag het even aan een ander.. Dus ja, ik heb er zelf, vrij veel dagen tussen dat ik alleen maar bij andere aan het bureau zit, om dingen uit te*

leggen, en dan te kijken, wat gaat hier mis. En ik zie dat ook binnen ons team dat ook heel veel gebeurd. Dus dit is een team effort. "

Respondent 2 indicates that people still fix issues on an individual basis: "*Dus niet Piet fixt het issue dat Jan heeft veroorzaakt, je zit hier met mensen die op vrijdag vrij zijn. Andere mensen die wel op vrijdag moeten werken..ja die zouden dan zeg maar.. Maar ik zie gewoon dat de build gewoon een weekend lang gewoon niet gefixt zijn. En dat wordt niet altijd als een issue gezien blijkbaar.*"

Weights

Table 9.6 - Motivation - weights

	Respondent 1	Respondent 2	Respondent 3	Respondent 4	Respondent 5	Respondent 6
Weights	-	5	4	4	5	4

The reason why respondent 3 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that the team effort is important so that the knowledge is also distributed across the team: *"Als je dat niet doet, dan krijg je dat mensen zich vereenzelvigen met de code en dan wordt het een koninkrijkje, ik ben ook een groot voorstander ervan, dat je de ene keer iemand een component onder handen neemt en dat het de andere keer door een ander wordt gedaan. Of in tandem he, dat je het dan samen doet. Maar dat die kennis ook verdeeld wordt over het team."*

The reason why respondent 2 thinks this CSF scores a 5 and is therefore seen as very important is because he/she also indicates that team effort, team responsibility and ownership contribute to the right DevOps culture: *"Maar met name het punt zeg maar, dat je het niet als een team oppakt. Maar dat het nog steeds als een individueel dingetje gezien wordt. Dat daar, dat vind ik zeg maar wel een gemis. Ik vind het niet passen bij.. Ik bedoel ik heb een idee erbij zeg maar, wat voor cultuur erbij Continuous Delivery en DevOps teams zou moeten.., wat in DevOps teams zou moeten.., en wat erin DevOps teams zeg maar uitgevoerd wordt, dat zie ik hier niet. Er wordt geen ownership, wij hebben, als voorbeeld, wij hebben gewoon producten die in productie draaien, waar niemand de vinger voor opsteekt, "Dat is van mij" letterlijk dus geen ownership. Ja, en responsibility dat heb je, maar dat moet je ook nemen, het wordt niet genomen. Maar het wordt blijkbaar ook niet door iemand geroepen van: "Maar dat is jouw verantwoordelijkheid!""*

The reason why respondent 4 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that a lack of motivation could possibly negatively affect other team members: *"Als er 1 of 2 mensen zijn die daar dat toch een beetje aan hun laars lappen. Dat je als heel het team laat bent, met het toch weer kunnen maken van een release. Als jij op donderdag weggaat, met in de veronderstelling, "Morgen is einde sprint, we hebben een release maandagmorgen". En je collega's hebben zoiets van: "Jaaa, is wel prima, ik heb ook bijna weekend" nou dan komt het er niet."*

Performance improvements

No performance improvements were observed.

Measurement of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the measurement of the CSF.

Implementation of the CSF
Due to a lack of time, the interviewer did not have enough time to ask about the implementation of this CSF.

Contextual relationships
It is observed that the substantiation of the weight indication for this CSF given by respondent 4 that "other team members may be negatively influenced by each other" might have a relationship with the performance requirement to "achieve the sprint".

Conclusion

This potential CSF could not be seen as verified CSF, because the CSF meets only criteria 1.

CSF - Resistance to change

Description of definition

Difficulty to change established organizational policies and cultures.

Clear definition

All respondents (5) indicate that the definition was immediately clear. Due to lack of time, at 1 respondent the interviewer did not have enough time to present this CSF to him/her.

Experience

All respondents (5) indicate that they have experienced this CSF.

Examples

Respondent 6 indicates that there is some kind of mismatch between the responsibility and authorization of team members of the DevOps team: *"Nou, wij zijn heel erg beperkt in onze mogelijkheden als het gaat om dingen in te richten op een EXP omgeving of een productie-omgeving. Nou kijk, wij hebben ook service-teams, bijvoorbeeld voor "integratie" die testen ook alles wat met queues enzo te maken heeft, of we hebben teams die gaan over de databases zeg maar. Onze TABer, onze technisch applicatie beheerder, heeft niet eens het recht om in de EXP database wijzigingen door te voeren. Dat moet het service-team doen die alles doet met databases. Ik vind dat service-teams, dat staat wel eens haaks op DevOps teams. Ik denk: "Ja, wij als DevOps team zijn daarvoor verantwoordelijk, voor onze hele applicatie, hoezo mogen wij dan, als het gaat om databases in EXP en productie, daar geen wijzigingen doorvoeren? Dat vind ik raar. Je hebt service-teams die gaan dan dwars door de DevOps teams heen en het ene service-team doet alles met betrekking tot queues, die mag van voor tot achter alles doen met queues, een ander mag alles doen met betrekking tot databases en ik vind dat wij op "ABL-gebied" ook alles zouden moeten kunnen doen met betrekking*

tot onze databases en onze queues enzovoort. Dus daar zit in mijn beleving, zit daar iets van een mismatch."

Respondent 4 indicates that security aspects such as hack testing do not cooperate conveniently with regard to Continuous Delivery because they still have to request hack tests and an inquiry usually takes about 6 - 8 weeks: *"We hebben nu nog dat we echt volledig hack-testen moeten aanvragen. Er vind een transitie plaats naar een licht-hack-test-regime. En dat houdt in dat je als DevOps team zeg maar gaat aantonen naar beveiliging: "Kijk wij hebben dat allemaal mooi geïntegreerd in onze pipeline", waar ik het net al over had he, die OWASP check bijvoorbeeld voor vulnerabilities en HP Fortify voor beveiligings dingen. Dat hoort allemaal bij die hack-testen, en als we als DevOps team gewoon kunnen aantonen, dat we dat iedere keer weer doen, iedere build. Nou dan kun je een vrijstelling krijgen voor zo'n hack-test voor een jaar zeg maar. En dat betekent dat je daarna gewoon, met andere wijzigingen voor die ene specifieke applicatie, snel die straat door kan en kun je dus eigenlijk buiten beveiliging om, ook de boel gewoon meteen naar productie mag brengen. En anders moest je altijd via beveiliging. We hebben hier nog soms een Change Advisory Board waar je ook langs moet, die dingen moet tegen kunnen houden. Ja dat werkt allemaal niet gunstig mee, met het Continuous Delivery verhaal. En ja om een voorbeeld daarvan te geven, ik denk een hack-test die moet je hier ook tussen de 6 en de 8 weken van te voren aanvragen. Maar je kunt nooit 6 tot 8 weken vooruit kijken, van zijn we dan ook echt zo ver, dat er ook iets te hack-testen valt."*

Respondent 2 indicates that there is a mismatch in the current organizational policy which causes a lack of application ownership: *"Ik had een voorbeeld over een applicatie in productie, wat al jaren draait, daar is dan vervolgens een issue mee. Maar dan is het verhaal van "Ja maar wie is daar eigenlijk eigenaar van?" En dan blijkt dus dat er gewoon geen eigenaar is"*

Respondent 2 also indicates that he/she experienced a lot of resistance when they wanted to break down the wall between the "infrastructure" side and the "software house" side: *"Wij hadden technische applicatiebeheerders, die een aparte vakgroep vormde, we werken bij het softwarehuis met test, bouw en ontwerp. En ondertussen is die technische applicatiebeheerders als 4e vakgroep bijgezet. Maar die leefde eerder aan de infrastructuur kant. De ICT was verdeeld in een softwarehuis-kant en een infrastructuur-kant, technisch applicatiebeheer zat aan de infrastructuur-kant, nou en van oorsprong als je de DevOps geschiedenis bekijkt dan gaat het ook over het weghalen van silo's. Van muurtjes, en daar zat een enorme muur tussen, wat dus gebouwd, getest, en ontworpen was dat werd letterlijk bijna over een muur geworpen naar een technisch applicatiebeheerder om gedeployed te worden op een server en dat is een van de eerste muurtjes die we ook hebben willen slechten en hebben kunnen slechten ook. Dat is een bewuste actie geweest. En daar zat dus veel, wel veel weerstand, wat dat betreft."*

The above example given by respondent 2 is also confirmed by respondent 4: *"We zijn eigenlijk bij het Softwarehuis begonnen met het introduceren van die DevOps teams, meer eigenlijk het waren in eerste instantie Dev teams, daarna is het een beetje doorgegaan naar DevOps in name only. Er waren wat mensen bij elkaar gezet, maar je merkte nog wel van: "Dit kan niet", vervolgens is het DevOps geworden. Maar dan nog hebben wij Opsers en niet iedere Opser heeft zoals wij dit hier noemen, een rijbewijs. En zolang ze dat rijbewijs niet hebben mogen ze ook niet naar productie brengen. En dat heeft best wel een hele tijd geduurd, voordat men uiteindelijk zover was dat het ook gewoon decentraal he, ook binnen een DevOps team, wel naar productie gebracht mag worden, en dat ook het beheer op productie ook daar lag. Het is heel lang eerst nog gewoon bij het oude clubje gebleven. Die vonden dat toch eng omdat weg te geven zeg maar."*

Weights

Table 9.7 - Resistance to change - weights

	Respondent 1	Respondent 2	Respondent 3	Respondent 4	Respondent 5	Respondent 6
Weights	-	5	3	5	4	3

The reason why respondent 6 thinks this CSF scores a 3 and is therefore seen as relatively important is because he/she indicates that their current way of working is not impossible either, it's just annoying, they have to ask other people to do things for them every time. But it's not pure need, they can live with it.

The reason why respondent 3 thinks this CSF scores a 3 and is therefore seen as relatively important is because he/she indicates that their own project is not progressing fast enough. *"Omdat ik met mijn eigen project dan niet snel genoeg voortgang heb. Ik heb er een hekel aan, om hier een dag te zijn, terwijl ik niet veel doe."*

The reason why respondent 2 thinks this CSF scores a 5 and is therefore seen as very important is because he/she indicates that by changing the organizational boundaries, "the technical application administrators" cannot be considered as a separate division anymore, but is now part of the DevOps team. *"De technisch applicatiebeheerders worden nu niet meer als een aparte club beschouwd buiten de DevOps teams, maar ze zitten gewoon in de DevOps teams. Dat heeft een wel een lange reis gehad, maar dat is gewoon gebleken dat dat gewoon een ontzettend succesvolle stap is geweest. De technisch applicatiebeheerder was verantwoordelijk voor het deployen van de applicaties op het applicatie-platform. Daar hebben we met het provisioning-portaal natuurlijk een heel groot deel van kunnen automatiseren waardoor die technisch applicatiebeheerders hun werk zeg maar makkelijker heeft kunnen doen, terwijl die zeg maar ook taken binnen het team, heeft kunnen oppakken. Ja daardoor echt een onderdeel van het team is geworden."*

The reason why respondent 4 thinks this CSF scores a 5 and is therefore seen as very important is because he/she indicates that you won't achieve the maximum result of Continuous Integration / Continuous Delivery if the DevOps team is not able to perform hack tests by themselves.

Performance improvements

According to respondent 4, performing hack tests themselves within the DevOps team would result in more speed, in order to go faster through the street:

"Nou dan kun je een vrijstelling krijgen voor zo'n hack-test voor een jaar zeg maar. En dat betekent dat je daarna gewoon, met andere wijzigingen voor die ene specifieke applicatie, snel die straat door kan en kun je dus eigenlijk buiten beveiliging om, ook de boel gewoon meteen naar productie mag brengen."

"Dan gaat het veel sneller ja."

Measurement of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the measurement of the CSF.

Implementation of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the implementation of this CSF.

Conclusion

This potential CSF could not be seen as verified CSF, because the CSF meets only criteria 1 and 2.

CSF - Complexity across customer organization boundary

Description of definition

No access to or control on a production environment or diversity and complexity of customer sites, which make it harder to fully automate the deployment process.

Clear definition

5 respondents indicate that the definition was immediately clear. Due to lack of time, at 1 respondent the interviewer did not have enough time to present this CSF to him/her.

Experience

3 respondents indicate that they have experienced this CSF. 2 respondents indicate that they do not have experienced this CSF. The reason why respondent 3 does not experienced this CSF is because he/she says that they always deploy their software on their own systems, they do not have an external customer.

Examples

Respondent 6 indicates that they have no access to production, which means they don't have the ability to analyze what's going on when something malfunctions: *"Wij hebben ook inderdaad geen toegang tot productie en wij hebben ook een zogenaamde EXP omgeving, dat is productie-like, ook daar heb ik geen toegang toe. Tenminste ik als ontwikkelaar zijnde niet. We hebben ook een technische applicatie beheerder, die kan daar wel deels bij. En dat maakt het inderdaad complex, om daar hele, om onze applicatie in een centrale [onverstaanbaar] te krijgen. Dat speelt nu, as we speak gewoon. Druk bezig om een EXP omgeving in te richten. En dan hapert er iets, en wij hebben niet de toegang en de mogelijkheden om goed te analyseren wat er aan de hand is."*

The above example given by respondent 6 is also confirmed by respondent 4: *"Ja, nou bijvoorbeeld met productie problemen. Als er gewoon ergens in productie iets niet goed gaat, en dan gaan wij in eerste instantie proberen om met onze eigen, in onze eigen omgeving, het na te bootsen of we het überhaupt voor elkaar kunnen krijgen. Maar als dat niet altijd lukt, dan zou het vele malen makkelijker zijn als jij gewoon in die productie omgeving kan kijken van: "Hee, hoe zit die database in elkaar? " is daar misschien iets geks mee? Wat staat er in de logging? Als ontwikkelaars zijnde,*

mogen wij dat sowieso niet. En de Opsers bij ons in het DevOps team, kunnen wel kijken, maar soms vaak ook niet overal en dan ben je toch wel weer, met hangende voeten gebonden."

Remarkably, respondent 2 indicates that he/she does not consider it necessary to have access to the production environment, because the environments are identical and they have tooling which can make extractions of the data so that they can simulate the production environment. *"Wij leveren niet op uiteindelijk aan ons eindpunt van onze Continuous Delivery proces is niet productie, maar een omgeving daarvoor. En die hele productie kant is dus voor ons niet beschikbaar. Wat ik al aangaf de data die daar staat, mogen wij om privacy redenen, mogen wij daar niet mee werken. We hebben dus ook wel tooling om inzicht te krijgen in wat er in productie speelt. Wij beschouwen alle omgevingen als productie omgevingen. Oke belangrijk punt. Dus wij zorgen dat er geen verschil is, kwa inrichting tussen of het nou de ontwikkelomgeving is of een testomgeving of waar je het ook voor gebruikt, die zijn allemaal identiek. De inhoud van de databases, die verschilt. En wij hebben tooling waarmee we extracties kunnen doen, dat we 1 op de 10 records pakken, of alles tussen die datum en die datum of nou eigenlijk dan hebben we daarnaast nog tooling, die anonimiseert, dus die de persoonlijke gegevens eruit filtert en daar bagger van maakt. En uiteindelijk hebben wij dus die hele productieomgeving ook niet nodig. Dus de voorzieningen zijn zodanig dat het dus een onbelangrijk aspect wordt."*

Weights

Table 9.8 - Complexity across customer organization boundary - weights

	Respondent 1	Respondent 2	Respondent 3	Respondent 4	Respondent 5	Respondent 6
Weights	-	1	-	4	-	1

The reason why respondent 6 thinks this CSF scores a 1 and is therefore seen as unimportant is because he/she indicates that the tooling does have access to production and other people within his team also have access. So it is unimportant to him/her to have access by them self. *"Ik vind het wel vervelend als er 1 keer, bugs geconstateerd worden en dat ik niet toegang heb tot bijvoorbeeld logging en dat soort dingen. Dat wordt dan wel een beetje vervelend. Maar dat staat op zich nog los denk ik van Continuous Deployment. Als de tooling er maar bij kan he. Wij als team zijn er wel voor verantwoordelijk. Maar daar zijn processen voor ingericht, je hebt in Jenkins een hele pipeline en je hebt XL-Release die dat allemaal deployed, en die tooling die kan dat doen, zonder dat ik er zelf bij kan. En de tooling die kan er wel bij. En als de tooling er maar bij kan, dan vind ik het prima. Dan hoef ik geen toegang te hebben tot productie. Maar andere mensen wel. In die zin als team, heb je natuurlijk wel uiteindelijk toegang."*

The reason why respondent 4 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that if they can do more on their own, they might be able to fix it faster. Now they depend on third parties to look into production: *"Aan de andere kant denk ik voor het Continuous verhaal. Vind ik het wel belangrijk, want hoe meer ik zelf kan, of een Opser in mijn team, des te sneller je het misschien kan oplossen. En je bent nu afhankelijk van derden."*

The reason why respondent 2 thinks this CSF scores a 1 and is therefore seen as unimportant is because he/she indicates that they do not consider production as the end point and have all the

conditions in place in such a way that they are also independent of it: *"Wij leveren niet op uiteindelijk aan ons eindpunt van onze Continuous Delivery proces is niet productie, maar een omgeving daarvoor. En die hele productie kant is dus voor ons niet beschikbaar. Wat ik al aangaf de data die daar staat, mogen wij om privacy redenen, mogen wij daar niet mee werken. We hebben dus ook wel tooling om inzicht te krijgen in wat er in productie speelt. Wij beschouwen alle omgevingen als productie omgevingen. Oke belangrijk punt. Dus wij zorgen dat er geen verschil is, kwa inrichting tussen of het nou de ontwikkelomgeving is of een testomgeving of waar je het ook voor gebruikt, die zijn allemaal identiek. De inhoud van de databases, die verschilt. En wij hebben tooling waarmee we extracties kunnen doen, dat we 1 op de 10 records pakken, of alles tussen die datum en die datum of nou eigenlijk dan hebben we daarnaast nog tooling, die anonimiseert, dus die de persoonlijke gegevens eruit filtert en daar bagger van maakt. En uiteindelijk hebben wij dus die hele productieomgeving ook niet nodig. Dus de voorzieningen zijn zodanig dat het dus een onbelangrijk aspect wordt."*

Performance improvements

No performance improvements were observed.

Measurement of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the measurement of the CSF.

Implementation of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the implementation of this CSF.

Conclusion

This potential CSF could not be seen as verified CSF, because the CSF meets only criteria 1.

CSF - Acceptance by customer

Description of definition

Adopting the practice of continuous releases. Customer perception of their involvement in development and customer behaviour. Domain constraints. Feature discovery.

Clear definition

4 respondents indicate that the definition was immediately clear. 1 respondent had a question and asked: "What exactly was meant by Continuous?" After clarifying the definition, the definition was clear to him/her. Due to lack of time, at 1 respondent the interviewer did not have enough time to present this CSF to him/her.

Experience

3 respondents indicate that they have experienced this CSF. 2 respondents indicate that they do not have experienced this CSF. The reason why respondent 4 do not experienced this CSF is because they've been working on a completely new application for a year now and there hasn't been a real production or release yet.

Examples

Respondent 6 indicates that they go to production every 3 weeks and the customer just accepts that. This happens outside office hours, in the morning between 6 and 7. According to the respondent, this is just a small moment, "*je doet een redeploy, en het spul werkt weer*". He/she also indicates that things will be different if, for example, the customer is out every afternoon between 2 and 3 because of a small feature that the customer doesn't even experience: *"Kijk, zolang dat continu 1 keer in de 3 weken is, en 1 keer in de 3 weken ben jij s'ochtends van 6 uur tot kwart over 6 niet bereikbaar, daar heeft de klant geen eens weet van, dus dat accepteert die wel. Maar als jij elke dag s'middags van 2 tot 3 eruit ligt vanwege, een of andere kleine feature die de klant niet eens ervaart. Ja dan wordt het natuurlijk anders."*

Respondent 3 indicates that in his/her current project they introduced Slack groups, which are a kind of message medium, in which they are in very brief contact with the customers of their services. Wishes and problems are then placed in such a Slack group and they can all respond to them. And make sure that the problem is solved. He/she indicates that this increases customer involvement and that the customers also accept this way of working: *"In het huidige project, hebben we op een gegeven moment, een aantal Slack groepen toegevoegd, wat ik net zei, voor zo'n machine koppeling daar gaan dus andere systemen aankoppelen. Interviewer: Maar een Slackgroep, is volgens mij een notificatie of push? Geïnterviewde 1: Ja een soort berichten medium. Waarin wij dus heel kort in contact staan, met de afnemers van onze diensten. Dat zijn XXXXXXX, dat zijn die bedrijven die die pakketten maken voor scholen. Binnen XXXXXXX gebruiken we een bepaalde omgeving die open staat naar buiten, de veldtest omgeving wordt dat genoemd, daar deployen we regelmatig heen, zodat zij ook hun testen kunnen doen. En als zij daar iets raars zien, dat kan zijn op informatiemodel-niveau van: "Hee, dat hebben we toen wel zo afgesproken, maar dat zou ik eigenlijk anders willen" of "Hij doet het niet" of "Hij reageert niet zoals ik heb verwacht." Dat wordt gepost in een Slack. En we hebben ook allemaal toegang tot die Slackgroep, en we kunnen dus heel kort daar ook allemaal op reageren. En zorgen dat het probleem opgelost wordt. Dus het is het, die customer involvement hebben we daarmee vergroot en deze klanten die accepteren het ook. Hoe het op die manier geregeld wordt. En ik zie dat op alle niveau's eigenlijk daar vragen in worden gesteld die de ene keer in het informatiemodel dingen laten wijzigen en de andere keer op implementatiemodel laten wijzigen, dat ze dingen op een andere manier moeten gaan maken."*

Weights

Table 9.9 - Acceptance by customer - weights

	Respondent 1	Respondent 2	Respondent 3	Respondent 4	Respondent 5	Respondent 6
Weights	-	-	5	-	5	4

The reason why respondent 6 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that it's important for the reputation of the organization: *"Je wilt niet als partij bekend staan, van: "Nouja, dat zijn die gasten, die website die ligt er heel veel tijd, ligt die eruit.""*

The reason why respondent 5 thinks this CSF scores a 5 and is therefore seen as very important is because he/she indicates that if the customer doesn't think it's important, his/her motivation will be gone for the most part: *"Als die er niet op zit te wachten, waarom zou ik dan grote inspanning leveren om iets heel snel op te kunnen leveren, terwijl de klant dat helemaal niet wil."*

The reason why respondent 3 thinks this CSF scores a 5 and is therefore seen as very important is because you make something for the customer, you want to have answered the customer's request properly and you can only do that if that line is short, that you also understand what the customer is struggling with and exactly know what he wants: *"Een aantal schermen gemaakt voor een beheerder, op een gegeven moment zei ik tegen die beheerder van: "Kom maar naast me zitten, ik ben hier nu bezig, en hoe wil je dit zien? " en dan heb je echt een hele korte lijn. Ik kan wel een schermpje bedenken, van dit zou volgens mij wel handig zijn. Maar iemand die het werk uitvoert. Die moet er mee werken. Dus die lijn moet gewoon kort zijn. En als iemand naast me zit, dan heb ik een hele korte release cycle, namelijk we doen veranderingen en je ziet gelijk wat er gebeurd."*

Performance improvements

No performance improvements were observed.

Measurement of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the measurement of the CSF.

Implementation of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the implementation of this CSF.

Conclusion

This potential CSF could not be seen as verified CSF, because the CSF meets only criteria 1.

CSF - Sales and intermediaries

Description of definition

When user data is not accessible due to intermediaries.

Clear definition

5 respondents indicate that the definition was immediately clear. Due to lack of time, at 1 respondent the interviewer did not have enough time to present this CSF to him/her.

Experience

1 respondent indicates that he/she has experienced this CSF. 4 respondents indicate that they do not have experienced this CSF. The reason why respondent 3 does not experienced this CSF is because he/she indicates that this doesn't play a role with them, that's why they using Slack, there's no intermediary between them.

Examples

Respondent 4 indicates that user data is not accessible from colleagues in The Hague: *"Ja, ook voor dezelfde applicatie die wij nu aan het maken zijn, komen er ook gegevens van onze collega's uit Den Haag. En dat zijn in dit geval een soort hulptabellen, waarin op basis van een postcode een voedingsgebied bepaald moet worden zeg maar. He, dat je zegt van: "Nou ik wil een school stichten in Groningen" dan maakt het even uit van is het een primair onderwijs, of is het voortgezet onderwijs. Bij primair onderwijs betekent dat zeg maar dat de doelgroep in een straal van 6 kilometer rondom die postcode zit, en bij voortgezet onderwijs is dat in een straal van 15 kilometer rondom. En op basis daarvan, moeten wij een heel lijstje tonen met plaatsen, van: "Oke, als je hem hier wilt stichten, dan zijn dit de potentiële plaatsen waaruit de leerlingen zouden kunnen komen, die interesse hebben voor een nieuw te stichten school." nou eigenlijk is dat een heel dom tabelletje. Daar zouden we rechtstreeks op in willen prikken in Den Haag. Maar dat mag niet. Dus dat wordt gewoon als een Excel bestandje deze kant op gestuurd. En dat moeten wij dan weer importeren in een hulp tabelletje. En dan denk ik: "Ja, en dan ben je toch 1 bedrijf"."*

Weights

Table 9.10 - Sales and intermediaries - weights

	Respondent 1	Respondent 2	Respondent 3	Respondent 4	Respondent 5	Respondent 6
Weights	-	-	-	5	-	-

The reason why respondent 4 thinks this CSF scores a 5 and is therefore seen as very important is because it takes them a lot of extra time and money to get the same result.

Performance improvements

No performance improvements were observed.

Measurement of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the measurement of the CSF.

Implementation of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the implementation of this CSF.

Conclusion

This potential CSF could not be seen as verified CSF, because the CSF meets only criteria 1.

CSF - Quality

Description of definition

Preserving quality and adequate documentation.

Clear definition

3 respondents indicate that the definition was immediately clear. 1 respondent had some questions, but after clarifying the definition, the definition was clear to him/her. Due to lack of time, at 2 respondents the interviewer did not have enough time to present this CSF to them.

Experience

All respondents (4) indicate that they have experienced this CSF.

Examples

Respondent 5 indicates that they work with JIRA issues in their work method. An issue will only be closed when it is resolved and accepted by the customer: *"Wij werken dan ook met issues he JIRA issues, in onze werkmethode. Dat een issue pas is afgehandeld, als de klant akkoord geeft, op de afwerking en de oplevering van dat issue. Dus die geeft zijn decharge eraan zeg maar. Dat zijn dan, dat is niet heel groot, maar dat zijn gewoon kleine dingetjes, maar ook voor kleine dingetjes moet, eigenlijk moet gewoon voor elk issue wat opgeleverd wordt, moet die akkoord op geven. Dus dat zit wel een beetje in onze werkwijze ingebakken."*

Respondent 6 indicates that they measure the degree of test coverage of the code to preserve quality coding: *"Ja, een voorbeeld als het gaat om kwaliteit van de code, nou dat wordt onder andere denk ik wel bepaald door.. Daar gebruiken we ook tooling voor die bijvoorbeeld meet wat de test-dekkingsgraad is van je code."*

Respondent 6 also indicates that there are all kinds of tools which can help you to ensure the quality of your code: *"Ja, OWASP scans worden gedaan he, of jij frameworks gebruikt, waar al bugs in bekend zijn, tegenwoordig heb je ook nog een black duck hub, daar is van alles voor. Die scannen ook de kwaliteit van je code, dus er zijn allerlei tools voor, SONAR, die scant jouw code en die kijkt naar, of je bijvoorbeeld variabelen hebt, die helemaal niet gebruikt worden."*

Respondent 4 indicates that the quality of the code with comments, is much more important than the functional documentation: *"Het voorbeeld wat ik wil noemen is nog uit de dakpansgewijze ontwikkelmethode, en daar was overigens de software kwaliteit gewoon goed hoor. Ik bedoel daar niks aan. Maar daar was het wel belangrijk om hele adequate documentatie te krijgen. Als het functioneel niet bijgewerkt wordt, en op een gegeven moment met reverse engineering vanuit de programmatuur moet gaan vertellen, hoe het waarschijnlijk in productie draait. Ja dan sla je de plank wel mis., Kijk en daar komt dan weer de andere kant van het DevOps verhaal bij, DevOps dat wil dat je dat wat sneller gaat doen, dus dat betekent dat je ook veel eerder met het hele team betrokken bent in het ontwikkelproces daarvan. En bij ons, geldt dat eigenlijk wel zo dat de code belangrijker is dan de documentatie daarvan. De kwaliteit van de code is belangrijker dan de documentatie van wat het stukje proces, waar je op dat moment mee bezig bent, wat het doet. Omdat het A. in de hoofden zit, en vaak ook wel in de issues genoemd is, van: "Als, product owner wil ik dat ik, een initiatief kan indienen" nou dat is dan zeg maar, een user story. Nou daar komen een aantal use cases bij. En in die*

use case beschrijving, want die wordt ook weer opgenomen in zo'n soort format. Wordt er iets meer gespecificeerd van wat er precies gedaan moet worden. Dus daar staat het in principe allemaal al in. Hoef je niet een hele uitgebreide functionele documentatie te maken. Voor eventuele oplossingen van bugs, kijk ik toch in de code. En daar wordt ook wel commentaar ingegeven. Dat vind ik belangrijker dan dat je hele uitgebreide functionele documentatie hebt. Waar bijna niemand meer naar kijkt."

The above example given by respondent 4 is also confirmed by respondent 3, he/she also indicates that he/she doesn't accept the review when there is no adequate documentation inside the code. *"Dan heb ik het met name over de documentatie binnen code, dat je aangeeft, wat iets hoort geacht te doen, zodat andere ontwikkelaars niet de code in hoeven duiken, om te kijken van: "Wat gebeurd hier eigenlijk?" en voor JAVA, wordt dat het JAVAdoc genoemd, en er is ook allerlei tooling voor omdat heel snel beschikbaar te hebben als je gaat ontwikkelen. En ja daar hamer ik toch wel op, als ik zie dat er methodes zijn, waar geen of weinig JAVAdoc bij staat, dan keur ik de review niet goed."*

The example given by respondent 3 implies that there are also "code reviews" to preserve quality.

Weights

Table 9.11 - Quality - weights

	Respondent 1	Respondent 2	Respondent 3	Respondent 4	Respondent 5	Respondent 6
Weights	-	-	4	4	5	3

The reason why respondent 6 thinks this CSF scores a 3 and is therefore seen as relatively important is because he/she indicates that quality is not always crucial, timely delivery for example, can be even more important. *"Aan de ene kant ben ik wel geneigd om altijd te gaan voor kwaliteit, om het beste zeg maar, ik wil ook graag die Ferrari maken, maar er zijn meer aspecten in het leven die van belang zijn, want kwaliteit is het allerbelangrijkste, als jij kwalitatieve, supergoede, mooi stukje software hebt gebouwd, maar je bent allemaal veel te laat met opleveren. En iedereen heeft dan zoiets van: "Naja, laat maar hangen, want het is te laat" , ja het is niet alles, dat zijn keuzes die moet je af en toe moet maken en de kwaliteit is niet altijd van doorslaggevend belang."*

The reason why respondent 3 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that adequate documentation provides speed, because other people don't have to dive into the code: *"Omdat je anders iets kunt maken, waarvan niet duidelijk is, wat je doel is geweest, toen je het maakt. Als ik een functie uit programmeer, dan is dat een functie, en die doet iets. Maar dat hoeft niet hetzelfde te zijn, als de reden waarom ik de functie heb gemaakt. Dus door te beginnen met de reden waarom ik de functie maak, en wat ik wil dat die doet, en daarna weer code te maken, kan je ook zelf zien van: "Hee, doet deze code ook wat die geacht wordt te doen?" en als dat "ja" is dan komt die door de eigen controle in, en hebben de andere gebruikers die jouw functies gaan gebruiken, gelijk: "Dit doet die, en dit is het resultaat en hij doet het op deze manier" hoeven ze dus de code niet in te duiken. Dat geeft gewoon versnelling. Want anders zie je alleen de naam van zo'n functie, en die hoeft niet, het is een lastige, je kunt wel hele lange namen gebruiken, en dan nog hoeft die niet beschrijvend genoeg te zijn om uit te drukken wat die functie precies doet."*

Performance improvements

According to respondent 3, adequate documentation provides speed:

"Hoeven ze dus de code niet in te duiken. Dat geeft gewoon versnelling."

Measurement of the CSF

Respondent 6 indicates that they measure the degree of test coverage of the code to preserve quality coding:

"Ja, een voorbeeld als het gaat om kwaliteit van de code, nou dat wordt onder andere denk ik wel bepaald door.. Daar gebruiken we ook tooling voor die bijvoorbeeld meet wat de test-dekkingsgraad is van je code."

Implementation of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the implementation of this CSF.

Conclusion

This potential CSF could not be seen as verified CSF, because the CSF meets only criteria 1 and 2. Criteria 3 could not be confirmed because no measurement report or document was provided to prove this.

CSF - Customer involvement

Description of definition

Preparing and receiving customer input, establishing a customer sample group, and delivering feature growth.

Clear definition

All respondents (6) indicate that the definition was immediately clear. At 1 respondent it was not clear if the definition was clear to him, but he/she mentioned an example directly so it was obvious that the definition was clear for him/her.

Experience

4 respondents indicate that they have experienced this CSF. 2 respondents indicate that they do not have experienced this CSF. Respondent 6 do not experienced this CSF and just indicates that he/she has experienced a lack of customer involvement: *"Ik vind het wel belangrijk, maar ik merk ook een gebrek daar aan, ik zou graag meer betrokkenheid willen van medewerkers. Het is moeilijk om ze te betrekken, want ja zij hebben het ook druk. En ja dat is wel lastig. Je geeft natuurlijk demo's. Daar krijgen we ook wel feedback. Maar dat ze nou super enthousiast zijn zeg maar, daar valt nog wel wat te winnen."*

This lack of customer involvement is also confirmed by respondent 1: *"Ja ik zou het graag willen…We geven elke 3 weken een demo, dan komen we.. wij maken een nieuwe applicatie, we hebben niet de klant die buiten XXXXXXX zit. Dus de leraren die een lerarenbeurs aanvragen. Daar hebben we nooit*

contact mee. Maar we hebben natuurlijk een groep gebruikers die met de applicatie moet gaan werken, en elke 3 weken geven wij een demo. We hebben geprobeerd dat ze ook de testen gaan doen, dat ze accepteren, dat zou je de sample groep kunnen noemen. Maar we krijgen geen input, ik heb nog nooit eigenlijk zo'n groep meegemaakt die eigenlijk hun hakken in het zand hebben, die zeggen: "Veranderingen?? Ik wil geen veranderingen!" Dat wordt ook gezegd, we hebben een bijzondere groep en die willen eigenlijk geen veranderingen. Behalve dan de applicatie die doet het natuurlijk niet goed die ze nu hebben. Ze moeten heel veel handmatig doen en daar klagen ze ook over, maar als we aankomen met nieuwe dingen, dan is het eerst van "Jaaa, nee! ingewikkeld!, moeilijk!" Het is heel jammer."

Respondent 1 also indicates that they have also developed a remedy to involve the customer, namely by doing click-sessions: *"Zij hebben het druk en nou ze hebben de tijd om alles te testen, en dan willen ze het niet. Dus we hebben ook klik-sessies nu. Dat is ook wel slim bedacht, dat doen we gewoon dus anders, en dan vertellen we eigenlijk wat ze moeten doen. Ze hebben dus een nieuwe applicatie, en dan zeggen we: "Nou, dit is een aanvraag, jij wilt het nu wijzigen en je wilt dat die in het buitenland gaat studeren., ga maar doen!" en dan gaat.. Dan worden ze wat enthousiaster van "Oh maar dit is een mooi…, dit ziet er mooi uit, ohh dit werkt handig!" in plaats van dat we de demo geven en dat ze zeggen: "Moeilijk! Moeilijk! En hoe moet het dan als dit het? En hoe moet het dan? " Nou kom maar op, en dan vragen we: "Oh hartstikke goed, beschrijf dan graag waar jij denkt dat het fout gaat of wat nu dan altijd fout gaat?" "Nee, daar heb ik geen tijd voor.." beetje frustrerend soms."*

Examples

Respondent 5 indicates that they use a "Sprint 0", in which the customer is present, so that the customer's wish can be clearly determined. *"In het begin moet je gewoon helder hebben wat de klant wil dat hebben wij een beetje geformaliseerd, wij beginnen met een Sprint 0. Je werkt in sprints he, in een project. Werk je in een aantal sprints, en in elke sprint lever je een stukje op. Soms heb je een projectteam geformeerd en dan is de klantvraag nog niet helemaal duidelijk. Dat zit er bij in, van: "Jullie moeten daar iets mee". Want de klant die wil er wel wat mee, maar de informatie moet nog wel gehaald worden. Daar gaan we sprint 0 voor bedenken. Dan gaan we dus met z'n allen die klant bevragen, "Wie is onze klant eigenlijk?", "Wat is het probleem?" "Wat is de wens?" en "Welk probleem lost dit op?" enzo, gaan we heel diep op in, met het hele team. En dan wordt er in sprint 0, wordt dat, moet dat helder zijn. En geformaliseerd en na Sprint 0, begint dan eigenlijk het echte ontwikkelen van het product."*

Respondent 3 indicates that an information model has been developed as a result of frequent consultation with stakeholders: *"Dus in het begin traject en nog steeds, is er overleg binnen het XXXXXXX wordt dat genoemd. Dat zijn de XXX-raad en…nou die dingen. En mensen daaruit, die kennis hebben van hoe met de XXXXXXX dingen omgegaan wordt, die overleggen zijn het begin heel frequent geweest, waaruit een bepaald informatiemodel is ontstaan, en die overleggen zijn er nog steeds omdat informatiemodel ook te toetsen, maar je ziet wel, de frequentie kan omlaag omdat het eigenlijk nu gestabiliseerd is. Ja, daar zijn hele discussies geweest..dat was echt van wat je noemt he, van mensen die daar echt bij betrokken zijn, die ook die features belangrijk vinden. Nou dat zag je daar heel duidelijk in terug."*

Respondent 2 indicates that Business Analysis Teams (BAT) are receiving customer input and feedback and provide this information to the DevOps teams. This is also their responsibility. They also create the user stories and are involved from the beginning of the process: *"In eigenlijk wel ieder team wordt deels aangestuurd he de opdrachten die uitgevoerd worden in het team, worden*

voorbereid door een BAT. (Bedrijfs-analyse-team). Waarin dus het klantperspectief meegenomen wordt. Maar ook helemaal in het begin van een traject, er wordt.., ik verwacht dat er een vorm van interviews is, bedrijfsanalisten / informatie-analisten, die zullen met gebruikers praten over hun wensen en uiteindelijk dat dat weer terugkoppelen richting.. In user stories richting de DevOps teams. Ieder team heeft een product owner als rol, die het team aanstuurt. Maar hij is samen met een senior uit het team, vertegenwoordigd in het BAT."

The above example given by respondent 2 is also confirmed by respondent 4: *"Ja, hij is zo ver dat we zeg maar met die demo's, dat we daar zeg maar de input krijgen van de klant, op het stuk wat we op dat moment gemaakt hebben, of dat is wat zij verwachten. En wat het zou moeten zijn..., even een stukje terug in het proces, daarvoor hebben wij een functioneel ontwerper en een UX-designer, die eigenlijk alleen maar de interactie met die klant hebben. En die klant is in ons geval bij XXXXXXX vaak een bedrijfsanalist of informatieanalist, die weer uiteindelijk contact met de eindklant heeft. Dus daar zitten best wel wat kansen op ruis zeg maar, op de lijn. Het liefste zou ik, als DevOps team, gewoon rechtstreeks contact hebben met die klant."*

Weights

Table 9.12 - Customer involvement - weights

	Respondent 1	Respondent 2	Respondent 3	Respondent 4	Respondent 5	Respondent 6
Weights	4	4	4	4	5	4

The reason why respondent 4 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that customer involvement implies speed: *"Omdat ik zelf met die klant, waarschijnlijk heel snel, sneller kan ontwikkelen. Die in minder stappen, het eindresultaat heeft, wat hij voor ogen heeft."*

The reason why respondent 3 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that it is important that the customer and the producer can work from the same image: *"Om juist de vrager en de maker bij elkaar te brengen, dat ze vanuit hetzelfde beeld werken. De vrager weet wat zijn de beperkingen bij het maken, de maker weet wat zijn de wensen exact bij de vraagkant."*

The reason why respondent 2 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that for example "the ability to deliver faster", improved the culture in the sense that the customer accepts that effort is needed that has to be invested in the technical side of continuous delivery, instead of telling the customer a technical story: *"Het gaat weer ook over cultuur. Het technische verhaal waar we het eerder over hadden, daar heeft een klant niet zo heel veel aan. Maar uiteindelijk krijgt die sneller, nieuwe features die die wenst of waarvan we hopen dat die die wenst en waar die blij van wordt. En het draagt dus bij tot "acceptance" van alle effort die we aan de technische kant erin steken, omdat zeg maar aan de management kant ook weer te belichten. Zeg maar van "kijk, onze klanten worden er ook blij van, het is niet alleen maar een technisch dingetje". Het verbeterd de cultuur. Zie het technische verhaal zeg maar waarmee ik sommige mensen kan overtuigen waarom ze het zouden moeten doen, dat gaat niet landen bij de klant. Dat boeit hem niet. Maar er zijn andere aspecten, waardoor het ook voor hun wel interessant is."*

The reason why respondent 1 thinks this CSF scores a 4 and is therefore seen as important is because he/she produces it for the customer: *"Ja ik maak het toch voor hun, het is heel simpel. Ik maak het voor iemand anders, die moet ermee werken en dat moet zo efficiënt en prettig mogelijk."*

The reason why respondent 6 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that it is important for the success of the project, you want to deliver an application which they are happy with: *"Ja dat is wel de belangrijkste, wij zijn niet een webwinkel ofzo, die klanten moet trekken, kijk een leraar, die wil graag een beurs. Dus die is sowieso al gemotiveerd om de dingen in te vullen. En als wij een heel gebruiksonvriendelijke applicatie maken, ja dat is dan vervelend, maarja, als die een kwartiertje extra nodig heeft, om zijn aanvraag te doen. En hij kan daarmee subsidie krijgen van een paar duizend euro, Mwuah dat doet die wel denk ik. Maar je hoeft niet klanten te trekken, dat is de andere kant van het verhaal. Maar als het gaat om de medewerkers-schermen, dan wil ik wel graag dat de medewerkers blij zijn met de applicatie."*

The reason why respondent 5 thinks this CSF scores a 5 and is therefore seen as very important is because he/she indicates that that you're probably making something that wasn't intended and therefore doesn't meet the quality standards, which could result in dissatisfied customers: *"Want als je die informatie niet goed haalt. Ja dan kan je wel raden wat er met de rest gebeurd. Haha, dan maak je waarschijnlijk iets wat niet de bedoeling was. Ja, en dat niet alleen, voldoet dan waarschijnlijk ook niet aan de kwaliteit die hij verwacht had. Dus dan krijg je ontevreden klanten dus, ja."*

Performance improvements

Respondent 4 indicates that customer involvement implies speed:

"Omdat ik zelf met die klant, waarschijnlijk heel snel, sneller kan ontwikkelen. Die in minder stappen, het eindresultaat heeft, wat hij voor ogen heeft."

Measurement of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the measurement of the CSF.

Implementation of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the implementation of this CSF.

Conclusion

This potential CSF could not be seen as verified CSF, because the CSF meets only criteria 1 and 2.

CSF - Test complexity & source code control

Description of definition

Aspects on automating, configuring and using test environments, and source code control. Due to higher demands as compared to traditional (not agile) ways of work.

Clear definition

All respondents (6) indicate that the definition was immediately clear.

Experience

All respondents (6) indicate that they have experienced this CSF.

Examples

Respondent 6 indicates that test automation is very important, because as soon as a new part of functionality has been created, you as a tester will never manually go through everything again: *"Maar als het gaat om het automatiseren van de tests zelf, testautomatisering, dat vind ik wel heel belangrijk. Sterker nog, dat was voordat we aan de herbouw zijn begonnen. Hadden we al geconstateerd dat er geen automatische tests waren, voor de oude applicatie. En toen hebben we nog een poging gedaan om achteraf dat alsnog te bouwen. Daar heb ik best wel energie ingestoken, uiteindelijk niet helemaal gelukt. Maar als je dat niet hebt, dan ga je meemaken dat je links iets aanpast, en dan valt er rechts iets om. Gegarandeerd gaat dat gebeuren. Want he, als ontwikkelaar, je ontwikkelt een stukje functionaliteit, dat wordt specifiek getest en dan gebeurd er misschien wel een algemene test, maar je gaat nooit als tester alles weer helemaal bij langs, dat kan ook allemaal nog net niet, en daar zijn die automatische tests dus heel belangrijk voor."*

Respondent 6 also indicates that Cucumber is used to test applications and test automation could be seen as a prerequisite for Continuous Integration and Continuous Delivery: *"Dat zijn integratie-testen, dat zijn schermen-testen, he dan ga je gewoon automatisch een bepaalde (in ons geval dan) aanvraag indienen en dan klik je allemaal door schermen heen, en daarna kijk je of de aanvraag, voor die gebruiker dan in zijn overzicht met aanvragen staat, je test of de brieven zijn aangemaakt, je test heel de applicatie door eigenlijk. Daar gebruiken we Cucumber voor. Maar, ja ook voor Continuous Integration en Continuous Delivery is dat van cruciaal belangrijk. Want hoe wil jij continu deliveren, als jij niet automatisch kan testen? Dat kan niet."*

Respondent 4 indicates that they write unit-tests on their own: *"Nou het begint natuurlijk al, met allereerst op het moment dat wij de code maken, zowel als client side als aan de backend kant, dat we zelf unit-testen schrijven, he zowel voor het stuk JAVA, als het stuk Angular, daarmee proberen we in ieder geval de units die wij opleveren, om die zo goed mogelijk getest te krijgen. En dan gaan we echt niet compleet functioneel testen, dat we alle verschillende paden bij langsgaan, we proberen daar zo'n 80% van de code wel te pakken."*

Respondent 4 also indicates that there is a seperate team called "test automation" which have made it relatively easy for them to build their own integration-tests and regression-tests: *"Daarnaast is het zo dat al onze backend code maar ook de schermen door onze functionele testers die in het DevOps team zitten, geautomatiseerd getest worden. Daar is een heel framework voor gemaakt. Daar hebben we een club voor, dat heet "Test automatisering" en daarmee kunnen ze op een vrij eenvoudige manier zelf hun hele integratie-test en regressie-test in elkaar zetten, waardoor bijna gewoon de hele applicatie geraakt wordt. En ook aangeeft van als je wat wijzigt, van: "he verrek, het gaat hier fout, nou waarom gaat het fout?" "ohjaa, ik zie het al! Mijn regressie-test is niet goed*

opgezet, want dit en dit en dit is veranderd en dat ben ik vergeten." Of "Nee, de test is wel goed, maar de code is niet goed" daar komt wat anders uit, als dat we volgens de test verwachten."

Respondent 3 indicates that the complexity lies in the management of test environments, which is caused by the amount of test environments: *"De complexiteit die ik zie, is dat we een aantal testomgevingen hebben, en die ook allemaal beheerd moeten worden. En zeker als je een testomgeving die je naar buiten toe beschikbaar maakt, wil je ook exact weten welke versies daarop staan. Zodat je ook kunt targetten van: "Hee, in die versie zit een fout, we moeten daar wat aan doen!" Maar dat levert wel heel veel complexiteit erbij, want ieder van die omgevingen moeten apart geconfigureerd worden en die configuratie moet zoveel mogelijk stabiel zijn, maar dat levert je wel heel veel beheer punten ook op. Dus de opser in een DevOps, binnen XXXXXXX hebben we dat op autorisatie-niveau nog gescheiden, als ontwikkelaar kan ik in bepaalde dingen wel op de productie-omgevingen kijken maar ik kan er geen dingen opzetten. Dus die Opser die heeft het daar soms heel druk mee. Met het inrichtingen van omgevingen, en dat wordt best wel complex."*

Respondent 1 indicates that because of the pressure on the team, they don't do source code control: *"Eigenlijk door de pressure, er moet op een gegeven moment iets klaar zijn en wij wel uitgebreide geautomatiseerde testen hebben, hebben wij eigenlijk niet de rust om elkaars werk te reviewen."*

Weights

Table 9.13 - Test complexity & source code control - weights

	Respondent 1	Respondent 2	Respondent 3	Respondent 4	Respondent 5	Respondent 6
Weights	4	2	4	5	4	3.5

The reason why respondent 6 thinks this CSF scores a 3.5 and is therefore seen between relatively important and important is because he/she considers the building of automatic test environments to be slightly important, because it might take a little more time, but they'll make it somehow. On the other hand he/she indicates the automation of tests to be very important, because he/she indicates that automatic testing is a prerequisite for Continuous Delivery: *"Want hoe wil jij continu deliveren, als jij niet automatisch kan testen? Dat kan niet."*

The reason why respondent 5 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that automatic testing can be considered as a prerequisite for Continuous Delivery: *"Ja, want het is gewoon een voorwaarde om Continuous Delivery te kunnen doen."*

The reason why respondent 4 thinks this CSF scores a 5 and is therefore seen as very important is because he/she indicates that because of the complex code these days, that it can't really be tested anymore without automatic testing. By use of automatic testing, testing becomes easier and faster: *"Omdat het tegenwoordig zoveel complexe code is, dat het eigenlijk zonder niet meer kan. Nou, je kunt er vanuit gaan dat het gewoon gebeurd. Als jij dat handmatig moet doen, ja dan kun je niets heel snel, alles weer opnieuw testen."*

The reason why respondent 3 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that without source code control it is inconvenient and not useful for cooperation: *"Source code control, als het er niet is, dan kun je er ook wel wegen voor bedenken maar het wordt*

wel heel erg lastig. Dus in die zin vind ik dat belangrijk. Als jij niet in een centrale repository je code onderbrengt, dan kan dat betekenen dat mensen niet goed kunnen samenwerken. En dat iedereen zijn eigen kopietje heeft bijvoorbeeld. En daar is opzich mee te leven hoor, daar kun je allerlei afspraken over maken, maar dat is niet handig"

Performance improvements

Respondent 4 indicates that use of automatic testing implies speed:

"Als jij dat handmatig moet doen, ja dan kun je niets heel snel, alles weer opnieuw testen."

Measurement of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the measurement of the CSF.

Implementation of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the implementation of this CSF.

Relationships

It is observed that the substantiation of the weight indication for this CSF given by respondent 3: "*Als jij niet in een centrale repository je code onderbrengt, dan kan dat betekenen dat mensen niet goed kunnen samenwerken.*" might have a mutual interdependence between Continuous Integration and DevOps.

Conclusion

This potential CSF could not be seen as verified CSF, because the CSF meets only criteria 1 and 2.

CSF - Coordination

Description of definition

Increased need for coordination between team members and multiple teams. Organizational structure. Co-locate by feature not discipline.

Clear definition

5 respondents indicate that the definition was immediately clear. At 1 respondent it was not clear if the definition was clear to him. He/she indicates that he finds the definition very difficult to understand because they have made adjustments in the organization, as a result the "increased need" is no longer experienced.

Experience

3 respondents indicate that they have experienced this CSF. 3 respondents indicate that they have not experienced this CSF.

Respondent 2 indicates that he/she has not experienced this CSF, especially the "increased need for coordination" because they made a different team composition: *"Mijn punt is dat, we hebben dus aanpassingen gedaan in de organisatie waarmee zeg maar een antwoord gegeven wordt op die "increased need for coordination" maar als je dus die organisatie al hebt, dan is het dus ook niet meer nodig, en speelt het dus geen rol. Het is juist de teamsamenstelling, en daarmee dus ook voor een deel de organisatie aanpassingen."*

The above experience given by respondent 2 is also confirmed by respondent 5, who indicates that this is caused by making DevOps teams. *"Ja maar, als je een DevOps team hebt, dan heb je dat toch al impliciet geregeld? Ja door het maken van een DevOps team! Dan valt er niks meer te coördineren, ze zitten nog bij elkaar. Dus ik zie het niet zo."*

Examples

Respondent 6 indicates that there is a need for coordination to align the version numbers of different functionalities used by different applications: *"Wij hebben nu ook bepaalde functionaliteiten nog van een andere applicatie, ja die moet al wel gerealiseerd zijn, voordat wij naar productie gaan, dus dat moet wel gecoördineerd zijn. En als dat niet gecoördineerd wordt, en wij gaan naar productie terwijl een ander nog niet naar productie is, ja dan heb ik een groot probleem. Versienummers bijvoorbeeld, kwa interfacing op bepaalde versienummer zit. Als dat niet met elkaar praat, ja…"*

Respondent 4 indicates that there is a need for coordination to align the technology behind the applications between the different teams: *"We hebben hier ooit een portaal zeg maar, waarop je moet inloggen op XXXXXXX en dat portaal daar draaien onze applicaties in, maar wij moeten, om in dat portaal te kunnen hangen. Moeten we ook een aantal beveiligingsdingen aanroepen. Zo moet er voor zakelijke klanten gelden dat ze een token hebben, voor studenten gelden dat ze met DigiD moeten kunnen inloggen en dat geldt natuurlijk ook voor debiteuren. Medewerkers hebben ook een inlogcode waarmee ze erin gaan. Maar alle techniek die daarachter zit moet wel afgestemd worden tussen de verschillende teams. Als je daar geen afstemming overmaakt, dan kom je er denk ik ook niet goed in te hangen."*

Weights

Table 9.14 - Coordination - weights

	Respondent 1	Respondent 2	Respondent 3	Respondent 4	Respondent 5	Respondent 6
Weights	2	4	3	4	3	5

The reason why respondent 6 thinks this CSF scores a 5 and is therefore seen as very important is because coordination is necessary to align the different applications, so they can communicate between them: *"Het is een landschap van applicaties, dus de ene applicatie moet praten met de andere applicatie, dat moet je met elkaar afstemmen. Kan niet anders."*

The reason why respondent 4 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that coordination is necessary to establish one uniform standard for software, which is a lot easier: *"De consequentie is dat als jij te weinig coördineert, je niet 1 bepaalde standaard krijgt waarmee je de applicaties in de betreffende portaal kan hangen. En als iedereen maar doet wat die*

wil, dan moet je zeg maar als ontwikkelaar van die portaal-applicatie moet je al die verschillende stekkertjes gaan lopen inbouwen. Het is natuurlijk veel makkelijker om gewoon 1 stekker mechanisme te hebben."

The reason why respondent 3 thinks this CSF scores a 3 and is therefore seen as relatively important is because he/she indicates that if there is no coordination, you get several people visiting another party for all kinds of things, and if you coordinate that, you can bundle that and ask those questions in a much more focused way: *"Nou, dan krijg je wel dat meerdere mensen uit hetzelfde project, bij die andere partij langsgaan voor allerlei dingetjes, en als je dat coördineert dan kun je dat bundelen en kun je veel gerichter die vragen stellen."*

The reason why respondent 2 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that by coordination you can act more like a team, in the sense of being able to take over tasks from each other: *"Om als team, met hoofdletter T te kunnen functioneren. Taken van elkaar over kunnen nemen, je bent T-shaped, we verwachten natuurlijk mediaan / basis mensen minder, Maar een Senior iemand die verwachten we van dat die T-shaped is. Dat betekent dat..maar dat geldt niet alleen voor bouwers, we hebben wel kwa skillset zeg maar zit je tegen bepaalde disciplines aan. En je kiest zelf welke discipline waar je in gespecialiseerd..en de stam dat is logisch, dat is jouw functie, maar daaromheen , links en rechts daar kun je natuurlijk dingen pakken. En in mijn geval ik ben op het moment dat ik in het CD-team zit, kom ik heel veel met operations dingen in aanraking, dus het technisch applicatiebeheer gebeuren. Nou, maar dat kunnen testers ook doen. Punt is dat men taken van elkaar over kan nemen, en welke taken dat zijn dat doet er niet zoveel toe. Het gaat er niet om dat je alle taken kan overnemen, maar een aantal taken. En daarmee ga je meer als team kunnen functioneren."*

The reason why respondent 1 thinks this CSF scores a 2 and is therefore seen as slightly important is because he/she indicates that with DevOps, everyone in the team is more responsible, including for coordination: *"Ik vind misschien gewoon, het moet gewoon natuurlijk zijn. Je bent gewoon een team, gezamenlijk doe je iets...Maar het hoeft niet bestempeld te zijn van dit is mijn functie want ik moet het coördineren...Bij DevOps ben je meer zelf verantwoordelijk vind ik"*

Performance improvements

No performance improvements were observed.

Measurement of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the measurement of the CSF.

Implementation of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the implementation of this CSF.

Conclusion

This potential CSF could not be seen as verified CSF, because the CSF meets only criteria 1.

CSF - Communication

Description of definition

Intra and inter team communication, the right communication tools, awareness and transparency.

Clear definition

All respondents (6) indicate that the definition was immediately clear.

Experience

All respondents (6) indicate that they have experienced this CSF.

Examples

Respondent 3 indicates that "Skype" is used as a communication tool to facilitate intra and inter team communication: *"Binnen XXXXXXX gebruiken we Skype en hebben we onze source code in een centrale repository staan waar iedereen bij kan. Dus je kunt heel makkelijk met mensen overleggen van: "Hee, ik zie dit!" die lijn is heel kort, dat wil niet zeggen dat niet iedereen altijd even blij is met Skype berichten. Maar het is ook heel makkelijk om vergaderverzoeken bij iemand in te schieten. De tooling is wel, de tooling is er om de communicatie goed intern, intern in het team en ook tussen teams goed te laten verlopen."*

Respondent 6 indicates that everyone in his/her team physically sit together and communicates very openly to each other and during the daily stand-up, they discuss what everyone is working on: *"Binnen ons team is het zo dat, ik vind het zelf dat wij een heel leuk team hebben. We zijn ook open naar elkaar toe, dus binnen het team wordt er volop gecommuniceerd over het werk, we doen ook sowieso al meer als de dagelijkse stand-up, dus op die manier weet je ook waar men mee bezig is. Dat is gewoon praten met elkaar, met waar je mee bezig bent." "Binnen het team wordt heel veel communiceert, we zitten als team ook bij elkaar. Heel bewust, en dat is ook goed. Fysiek bij elkaar ja. Kwa transparantie, het enige wat er misschien wel eens aan mankeert, dan denk ik, dat je elkaar aanspreekt op dingen he. Als je bijvoorbeeld inschat dat iets een dag duurt en je constateert dat een collega daar al 3 dagen mee bezig is. Dat je diegene erop aanspreekt van: "Hey, waarom ben je daar zolang mee bezig?" En dat hoeft helemaal niet perse aanvallend te zijn, maar ja dat is nog wel eens lastig."*

Respondent 6 also indicates that JIRA is used to create stories and to watch/trace issues: *"Ja en JIRA, wordt heel veel in gedaan. Nou stories waar we mee bezig zijn, andere teams leggen daar ook hun stories in vast. Dat is allemaal open voor iedereen zeg maar. Sowieso kwa lezen ja, en je kunt ook zo'n issue dan watchen. Als wij bijvoorbeeld een afhankelijkheid met een ander team hebben, en wij weten dat zij een bepaald issue daarvoor hebben, dan gaan wij die watchen, om te zien of daar beweging in zit."*

Respondent 4 indicates that the scrumboard is also often present on paper, so they can see what each team is doing and what its current status is: *"Nouja, wat wij aan communicatie hebben is dat wij bij de teams ook zeg maar op de plaatsen waar wij meestal de daily scrum hebben, ook vaak gewoon het scrumbord op papier hebben staan. Daarmee kun je gewoon van ieder team zien waar ze mee bezig zijn en hoe het er voor staat."*

Respondent 1 indicates that there is also a team which is called the “data warehouse”, which acts as an intermediary between teams who need data from each other: *“Nou het gegevensmagazijn is een hele leuke club, je zou kunnen zeggen, eigenlijk ben je gewend, wij zijn applicatie X en wij willen iets van Y hebben, dan heeft Y, die heeft een service en dan kunnen wij vragen: Geef mij even die gegevens, en dan hebben wij die gegevens. Dat kan, maar stel voor dat ik nou uit A,B,C,D,E,F iets moet hebben. Dan moet ik dus aan allemaal een call doen en al die calls moet ik samenvoegen en dan heb ik eigenlijk de gegevens die ik nodig heb. Nou dat is niet echt heel efficient. Dus we hebben een gegevensmagazijn en die zeggen “Okee, wij kunnen uit A,B,C,D,E en die Y kunnen we gewoon in de database, die gegevens uithalen. Dus je hebt 1 call die je aan ons doet, en wij halen het overal vandaan.”*

Weights

Table 9.15 - Communication - weights

	Respondent 1	Respondent 2	Respondent 3	Respondent 4	Respondent 5	Respondent 6
Weights	5	4	3	4	5	4

The reason why respondent 6 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that it is important to know from each other what everybody is doing: *“Nouja, je moet gewoon van elkaar weten waar je mee bezig ben, anders kun je volledig langs elkaar heen...Nou, ik vind denk de transparantie niet het allerbelangrijkste, maar wel dat je gewoon kan weet, waar je mee bezig bent. Dat je naar andere teams toe gaat, als je wat nodig hebt. Je kan wel een beetje in je uppie blijven proberen, maar dat helpt niet. Je moet elkaar op zoeken.”*

The reason why respondent 4 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that without communication, you don’t know when something is finished which is built by another team, where you are waiting for: *“Omdat je zonder communicatie eigenlijk nergens komt...Dat je gewoon op sommige vlakken gewoon niet weet wat je moet doen, wat er van je verwacht wordt, wanneer jij van een ander team kan verwachten dat er iets klaar is, waarop jij aan het wachten bent zeg maar.”*

The reason why respondent 3 thinks this CSF scores a 3 and is therefore seen as relatively important is because he/she indicates that transparency can serve to improve the team: *“Nou, omdat ik het liefst, graag wil en dat is dan met name de transparancy, dat je benaderbaar bent door anderen. En ook dat je open kunt zijn, over de dingen die je doet en binnen het team betekent dat je ook kunt communiceren zonder dat dat gevolgen voor jou heeft. Als je vind dat binnen een bepaalde sprint bijvoorbeeld iets niet goed is gegaan, of dat iemand iets niet goed heeft gedaan. Dan vind ik dat je dat gewoon moet kunnen zeggen. Met als doel, het team te verbeteren. Het is dan niet zo dat je zegt van “Jij hebt deze sprint, dat en dat soort dingen niet goed gedaan” Het zijn meestal de negatieve dingen die dan lastig worden om te benoemen. Ik vind dat je die moet kunnen benoemen. In de context ter verbetering. En niet om iemand af te branden.”*

The reason why respondent 2 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that communication tools like JIRA can support to realize the coordination in the team: *"Wij zijn op een gegeven moment, JIRA gaan gebruiken. Maar dat valt bij ons onder de categorie "Collaboration". Team collaboration tools. En met name JIRA natuurlijk speelt een enorme grote rol in het succes van Continuous Delivery. Je krijgt overzicht over je werkpakket en wat de stand van zaken daarvan is." "Omdat het bij elkaar zetten in een team, dat ging voor de coördinatie, het vervolgens in de praktijk dat waar maken, daar heb je gewoon ondersteuning bij nodig en dat gebeurd doormiddel van die tooling."*

Performance improvements

No performance improvements were observed.

Measurement of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the measurement of the CSF.

Implementation of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the implementation of this CSF.

Conclusion

This potential CSF could not be seen as verified CSF, because the CSF meets only criteria 1.

CSF - Knowledge and training

Description of definition

Sufficient proficiency, knowledge, skills and experience.

Clear definition

4 respondents indicate that the definition was immediately clear. At 1 respondent it was not clear if the definition was clear to him. He/she indicates "*yes and no*".

Experience

3 respondents indicate that they have experienced this CSF. 2 respondents indicate that they have not experienced this CSF. Respondent 5 indicates that he/she does not experienced this CSF, because he/she does not see the lack of skills: *"Maar ja, als programmeur, nee ik zie niet dat je voor Continuous Delivery andere skills nodig hebt, dan voor een gewone programmeur."*

Examples

Respondent 6 indicates that knowledge is definitely available within his/her team: *"Training niet zo zeer. Maar wel kennis als het gaat om dat er vanuit ons team zelf, bijvoorbeeld de pipelines inrichten op Jenkins, en als ze dat niet weten, dan zoeken we het uit, of gaan we bij collega's te rade, maar niet*

echt dat we daar trainingen in gevolgd hebben. Als die kennis er niet is dan ga je hem halen. Ofwel bij collega's, voorbeeldprojecten en Google uiteraard."

Respondent 4 indicates that he/she is busy with a POC and following workshops to experiment with certain web technologies in order to decide if this is interesting for the case organization: *"Ik ben op dit moment bezig met een POC voor low-code, wij ontwikkelen nu op dit moment alles in Angular en JAVA dat zijn [onverstaanbaar] webapplicaties, wij hebben nog heel veel legacy systemen. Als we die ook op die manier zouden moeten ontsluiten en opnieuw zouden moeten bouwen, nou dan zijn we nog wel even bezig zeg maar. En dat low-code verhaal dat is eigenlijk een soort 4e generatie-taal. Die op basis van het datamodel zeg maar heel veel dingen al voor je automatisch aan kan genereren. Dus simpele CRUD applicaties en dat soort dingen allemaal, je toetst je datamodel in elkaar, ik heb die en die velden in een entiteit, naja orderregel, order, order. regel, als je zo'n soort relatie hebt. Dan zeg je: "Oke, dit is m'n datamodel en ik wil zo'n soort scherm eroverheen" Het is meer het sleur en pleur. Nouja dat is allemaal net in de kinderschoenen, dus daar zijn we nu mee bezig om te kijken, zou dat wat voor XXXXXXX kunnen zijn. Naja, we hebben dan van 2 van dat soort systemen dan, heb ik een dag een workshop gehad zeg maar, dat je in een notendop leert kijken hoe het daarmee zit. Voor de rest proberen we dat dan zelf weer verder onder de knie te krijgen. Weliswaar met iemand van die bedrijven die hier dan een dag in de week meelopen om mee te helpen. Puur ook om te kijken, van: "Zou het wat kunnen zijn, om de rest van onze applicaties, die we nog niet omgebouwd hebben, maar bijvoorbeeld van die oude groene schermen die van een AS400 zijn, of we het daar ook mee kunnen doen""*

Respondent 3 indicates that the case organization established competence groups in order to absorb knowledge and to propose new things: *"Je ziet dat mensen in een team gewoon verschillende kennis-diepte hebben van dingen. Dus ja, er moeten medewerkers zijn die daar de kennis van hebben. Dan wel die die kennis willen opdoen, dus daarin willen gaan verdiepen, en dan ook die kennis gaan overdragen aan anderen...Binnen XXXXXXX zijn we continu in beweging van: "Op welke manier wij software deployen, met welke frameworks we die dingen maken." Er zijn een aantal groepen die zich daarin verdiepen, van: "Dit lijkt ons de beste keuze.' dat zijn competence-groepen... Dat is meestal op vrijwillige basis dat je daarin deelneemt, als je zegt ik heb affiniteit met dit, he dan kun je in die groep stappen, en ga je gezamenlijk met die groep dingen voordragen. He.. we zien nu ook dat er een.. vanuit XXXXXXX gevraagd is, van: "We willen veel meer naar een cloudoplossing toe", dat betekent voor software ontwikkeling dat je natuurlijk dat in bepaalde frameworks gaat doen. Zo'n framework gebruik je niet zomaar. Dus daar moet iemand zich wel in verdiepen. Die moet het kunnen uitleggen van: "Hoe werkt dit en hoe werk je ermee". Dus dat is belangrijk dat je die mensen hebt, en elk geval dat ze benaderbaar zijn, en dat je ook die kennis wilt opzuigen."*

Respondent 3 also indicates that information transfer also happens at the desk and sometimes there are pizza sessions: *"Binnen het team doen we dat gewoon door bij elkaar aan het bureau te gaan zitten en die informatie over te dragen. Binnen XXXXXXXX he van wat ik net zei, meer die cloudtools doen we af en toe pizza-sessies dan worden de dingen gepresenteerd, soms is dat bij een werkoverleg, dat er zo'n sessie wordt gepland om dingen uit te leggen. En die zijn heel nuttig. Best belangrijk dat dat gebeurd."*

Weights

Table 9.16 - Knowledge and training - weights

	Respondent 1	Respondent 2	Respondent 3	Respondent 4	Respondent 5	Respondent 6

Weights	-	-	4	4	-	3

The reason why respondent 6 thinks this CSF scores a 3 and is therefore seen as relatively important is because he/she indicates that not everyone in the team should know exactly how everything works, but he/she indicates that they are making use of a service team which carry out the deployment of application servers which are being used by the team. However, by working together with a service team, knowledge is kept away: *"Nou, niet iedereen binnen het team hoeft precies te weten hoe dat allemaal in elkaar steekt zeg maar, en dat is bij ons ook praktijk trouwens. Er is 1 of 2 mensen die hebben zich daar in verdiept. Hoe zit dat nou allemaal met die Jenkins pipelines, om de hele build cycle vorm te geven. En een ander neemt dat gewoon voor waarheid aan. Het werkt. Als ik iets commit dan gaat de hele molen draaien zeg maar.." "als het gaat om bijvoorbeeld de applicatieserver waarop wij deployen. Ja, zij regelen dat voor ons. En nouja zolang het goed gaat, gaat het goed, maar het heeft ook weleens een nadeel hoor. Dat is ook wel eens, er wordt ook wel..op de achtergrond doen zij dingen voor ons, waardoor kennis bij ons niet bewust, maar wel weggehouden wordt he, wij weten niet alles inhoudelijk van JBOSS applicatieserver, hoe dat precies zit met zo'n connectionpool en waar zit die configuratie nou precies. Ja dat wordt allemaal door XL-Deploy voor ons op de achtergrond aangemaakt. En ja, het zal wel. Het gaat goed. Nou prima."*

The reason why respondent 3 and respondent 4 thinks this CSF scores a 4 and is therefore seen as important is because they indicate that when you don't maintain your area of expertise, then at some point you will not be part of the game anymore:

Respondent 4:
"Nouja als jij geen voldoende adequate training krijgt in de toolset waarmee je je werk moet gaan doen. Of waarmee je bezig bent, dan.. Of je onderhoud je vakgebied niet, dan lig er op een gegeven moment gewoon uit, klaar."

Respondent 3:
"Ze zeggen wel eens, stilstaan is achteruit gaan. Als je je niet blijft ontwikkelen hierin, dan kun je op een gegeven moment niet meer mee."

Performance improvements

No performance improvements were observed.

Measurement of the CSF
Due to a lack of time, the interviewer did not have enough time to ask about the measurement of the CSF.

Implementation of the CSF
Due to a lack of time, the interviewer did not have enough time to ask about the implementation of this CSF.

Conclusion

This potential CSF could not be seen as verified CSF, because the CSF meets only criteria 1.

CSF - Tooling

Description of definition

Maturity of the tools and their surrounding infrastructure that sufficiently support the Continuous Integration, -Delivery/-Deployment process, including security, access controls and consensus among users about the choice of tools. Heterogeneous programming languages, operating systems and communication tools.

Clear definition

5 respondents indicate that the definition was immediately clear. Due to lack of time, at 1 respondent the interviewer did not have enough time to present this CSF to him/her.

Experience

All respondents (5) indicate that they have experienced this CSF.

Examples

Respondent 6 indicates that there is a list of tools which are being used to support the Continuous Delivery process. “IDE”, “Intelie”, “Jenkins”, “XL-Deploy”, “XL-Release”, “OWASP” and “Fortify”. He/she also indicates that some tools are very strict, when it comes to scanning of the code, to check if there are bugs in it, like Fortify, which causes a lot of false positives and therefore takes a lot of time: *“Er is natuurlijk een hele waslijst aan tooling, die hier bij XXXXXXX ook gebruiken. Sowieso natuurlijk je IDE, we gebruiken Intelie, maar ook voor Continuous Delivery, Jenkins, waarin we Jenkins pipelines hebben gedefinieerd. XL-Deploy, waarmee wij deployen. XL-Release, waarmee we releasen... Dus er is een hele, en tooling voor de beveiliging, had je er ook bijstaan geloof ik. Dus OWASP, security scans en dat soort dingen meer... ik heb nog wel eens het idee, dat sommige tooling als het gaat om het scannen van de code of daar ook bugs inzitten, zoals Fortify. Die zijn soms wel eens heel strict. Maar dat gaat dan misschien meer om inrichting. Soms dan denk ik wel eens, ik ben haast meer tijd kwijt aan het fixen van false positives, dan dat ze mijn code verbeteren. Dus ik vraag me wel eens af, of het helemaal kwa balans, goed komt. Maar over het algemeen, ja de tooling is volwassen genoeg.”*

Respondent 4 indicates that he/she helped with the development of a provisioning platform when he/she was part of the “Continuous Delivery” team, which is being used for the provisioning of develop-silos for applications: *“Ik heb een poos bij het Continuous Delivery team gezeten, het CD-team hierzo, dat is zeg maar het team dat hier de infrastructuur onder andere ook ... Voor het Continuous Delivery proces en die beheren dat, ik heb eerder in het interview heb ik het gehad over die verschillende webservers die wij gebruikten he, als ontwikkelaar [onverstaanbaar] en aan het eind WebSphere. Daar wilden we op een gegeven moment vanaf, we wilden voor de hele keten, dezelfde webserver hebben. Het duurde ook heel erg lang voordat er zo’n webserver ingericht was. En als die eenmaal ingericht was, was het ook maar de vraag of die zeg maar in de FAT omgeving, exact hetzelfde ingericht was als in de GAT omgeving. En daarna komt dan bij ons de exploitatie omgeving en uiteindelijk productie. Nou bleek bij de overgang van de ene naar de andere nog wel eens wat fout te gaan, het is allemaal handwerk. Zo gezegd als XXXXXXX zijnde, dat willen we niet meer. Daar willen we wat voor neerzetten. Nou daar hebben we met een externe leverancier en een aantal interne*

medewerkers is daar, iets voor geschreven, waarmee we nu als ontwikkelaar zeg maar, met ons schermpje kunnen aangeven van: "Ik wil een nieuwe ontwikkel-silo hebben voor een applicatie, en ik wil daaruit 2 JBOSS instances aan hebben, en gekoppeld worden met die en die database. [onverstaanbaar]" Nou dat vragen we dan aan, druk op de knop, en 5 minuten later staat een volledig ingerichte applicatieserver voor je klaar. Kun je je eigen applicaties opzetten en die kun je gewoon testen. Maar je weet ook gewoon dat, alles wat op deze server staat, dezelfde soort gelijke server aanvraagde voor de FAT omgeving en de GAT omgeving, dat die iedere keer op exact dezelfde wijze aangemaakt wordt."

Respondent 3 indicates that he/she sometimes has questions about the maturity of the tools they use, according to him/her the interfacing of "XL Deploy" is not that handy: *"Dan heb ik wel eens mijn vraagtekens bij de volwassenheid van de tools die we gebruiken. Met name 1 tool die gewoon kwa interface niet handig is, dat XL-Deploy ding, waar enorme lange lijsten in staan nu, zeker dat ontwikkel, dan heeft iedereen allemaal dingetjes echt aangemaakt voor zichzelf. Dus in plaats van wat je op productie hebt, heb je dan 1 ding. Heb je bij ontwikkeling in eens er 10 van staan. Plus dat de tool vaak gewoon omvalt. Maar de infrastructuur die is voldoende stabiel. De tooling is dus, dat ik denk van: "Ja, uhh vind ik niet echt prettig" en er is ook voldoende support."*

Weights

Table 9.17 - Tooling - weights

	Respondent 1	Respondent 2	Respondent 3	Respondent 4	Respondent 5	Respondent 6
Weights	4	-	5	5	5	5

The reason why respondent 6 and respondent 5 thinks this CSF scores a 5 and is therefore seen as very important, is because they indicate that tooling can be seen as a prerequisite for Continuous Integration, Delivery and/or Continuous Deployment:

Respondent 6:
"Ja, echt een randvoorwaarde ja."
Respondent 5:
"Zonder dat, kan ik mijn werk niet doen."

The reason why respondent 3 thinks this CSF scores a 5 and is therefore seen as very important is because he/she indicates that tooling is also a prerequisite especially to get enough support to implement this in the organization: *"Je kunt wel allemaal ideeën hebben, maar als je tooling niet goed is, dan krijg je het er nooit door. ...Voor CD/CI is die heel belangrijk...dan valt je hele Continuous Integration / Delivery / Deployment om, dan kun je het hele stramien niet volgen, want die tooling is brak. Die tooling moet zo stabiel zijn, dat je er op kunt leunen. Dat je zeker weet dat het werkt. En het moet toegankelijk zijn, dus het moet niet te complex zijn om die interface… het moet niet heel veel werk zijn... Ja, dus voor Continuous...voor dit project specifiek vind ik het heel belangrijk dat je goede tooling hebt, anders krijg je nooit voldoende draagvlak ervoor, om het door te voeren."*

The reason why respondent 1 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that the tooling takes care of the "deployables", which saves him/her a lot of time:

"Het neemt voor mij veel werk uit handen en zou ik verantwoordelijk zijn voor elke handmatige deployment of wat dan ook of de omgeving in de gaten houden dan kost mij dat gewoon veel meer tijd, wat ik niet kan besteden aan het ontwikkelen van software….Ja. Het wordt ook, het is zeg maar ook eenvoudiger.. Vroeger was het.. Hoefde ik het ook niet te doen. Maar ze kwamen wel bij je met "Wat moet ik hier invullen?" en "Waarom zou die dat niet doen?" Nee, dat is nu niet meer zo want ik heb een deployable opgeleverd en die gaat gewoon naar die omgeving en dan [onverstaanbaar]... dus die ga je gewoon stap voor stap door."

Performance improvements

No performance improvements were observed.

Measurement of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the measurement of the CSF.

Implementation of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the implementation of this CSF.

Conclusion

This potential CSF could not be seen as verified CSF, because the CSF meets only criteria 1.

CSF - Pace

Description of definition

Improving the speed of delivering deployments to the customer.

Clear definition

5 respondents indicate that the definition was immediately clear. Due to lack of time, at 1 respondent the interviewer did not have enough time to present this CSF to him/her.

Experience

3 respondents indicate that they have experienced this CSF. 2 respondents indicate that they have not experienced this CSF. The reason why respondent 6 indicates that he/she did not experience this CSF is because he/she usually release every 3 weeks in a continuous flow, so it is impossible to experience an improved speed of delivery: *"Nee dat heb ik niet meegemaakt. Want we releasen gewoon 1 keer per 3 weken. Het kan wel zijn dat er in 1 release, meer functionaliteit zit. Dus het releasen anzich gaan niet sneller, maar het kan wel zijn dat er meer inzit."*

Examples

Respondent 4 indicates that the speed of delivering deployments to the customer is increased by releasing (incremental) new functionality every 3 weeks, by comparing this example with a method

from the past: *"Kijk vroeger, bij het dakpansgewijze, was je gewoon 2 a 3 maanden aan het ontwerpen, en daarna was je een paar maanden aan het bouwen en werd het vervolgens een keer getest, en was je een jaar verder. En als je geluk had, was het datgene wat de klant wilde, en nu ben je gewoon in brokjes van 3 weken steeds wat aan het maken. Je ontwerpt het, je bouwt het, je test het, en je zet het klaar voor die klant. En die klant kan er in principe wat mee. Dat wil niet zeggen dat die net als vroeger, alles in 1 keer kan. Maar iedere keer zorg je ervoor dat hij of zij, weer een beetje extra functionaliteit erbij heeft. Waar hij ook meteen mee aan de slag kan."*

The example given by respondent 4 is also confirmed by respondent 5:

"Een voorbeeld is, dat je nu veel sneller producten oplevert. Ja daar zit nog een klein addertje onder het gras dan, want in de watervalmethode, breng je hele, hele grote brokstukken breng je in 1 keer naar productie. Dat gaat sowieso dus langzaam. En met deze methode, kun je ook steeds kleine stukjes, incrementeel naar productie brengen."

Weights

Table 9.18 - Pace - weights

	Respondent 1	Respondent 2	Respondent 3	Respondent 4	Respondent 5	Respondent 6
Weights	-	-	2	2	5	-

The reason why respondent 3 thinks this CSF scores a 2 and is therefore seen as slightly important is because he/she indicates that the tooling provides sufficient speed, which fits inside the customers time windows: *"Dat de tooling, die we nu hebben, ons voldoende snelheid geeft, om de dingen te doen binnen tijdframes die ook de klant heeft."*

The reason why respondent 5 thinks this CSF scores a 5 and is therefore seen as very important is because he/she indicates that in comparison with a method from the past, the increased delivery speed may have influence on the customer satisfaction: *"Nou, klanttevredenheid. Interviewer: Jij denkt dus, dat hoe sneller je levert, dat dat invloed heeft op de klanttevredenheid? Geïnterviewde 1: Ja, dat denk ik wel."*

The reason why respondent 4 thinks this CSF scores a 2 and is therefore seen as slightly important is because he/she indicates that the quality of the software takes precedence over speed: *"Omdat snelheid niet alles is, ik denk dat die meer gebaat is bij goede software die iets langer duurt. Als slechte software die heel snel geleverd is. Dus hij krijgt iedere, bij ons duurt een sprint 3 weken in ons DevOps team, in de 3 weken krijgt die wat. Nou ik heb liever dat die iedere 3 weken iets goeds krijgt, als dat die iedere week iets slechts krijgt."*

Performance improvements

Respondent 5 indicates that changing the delivery speed to a shorter incremental release schedule implies speed:

"Een voorbeeld is, dat je nu veel sneller producten oplevert.

Measurement of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the measurement of the CSF.

Implementation of the CSF
Due to a lack of time, the interviewer did not have enough time to ask about the implementation of this CSF.

Contextual relationships
It is observed that the substantiation of the weight indication for this CSF given by respondent 4 that "the quality of the software takes precedence over speed" might have a relationship with the performance requirement to "achieve the sprint".

Conclusion

This potential CSF could not be seen as verified CSF, because the CSF meets only criteria 1 and 2.

CSF - Pressure

Description of definition

Increased pressure on the team to have code ready to be deployed immediately, and too much transparency which causes customers to interfere with developers' work.

Clear definition

4 respondents indicate that the definition was immediately clear. Due to lack of time, at 2 respondents the interviewer did not have enough time to present this CSF to them.

Experience

3 respondents indicate that they have experienced this CSF. 1 respondent indicates that he/she has not experienced this CSF.

Examples

Respondent 6 indicates that he/she has experienced pressure because of deadlines, which forced them to make concessions to quality, which was the expense of code reviews and duplication of code at one place: *"Dat eerste deel dat gaat over druk om op te leveren zeg maar. Wij hebben nu die deadlines. En dat ervaren wij ook wel als druk, dan moet er gewoon wat klaar zijn, en dat legt wel druk op ons in die zin, dat we gewoon concessies moeten doen aan kwaliteit….ABL moet straks naar productie. Dat is gewoon zo. Dus dat is gewoon een keiharde deadline. En daardoor moeten wij nu af en toe keuzes maken, die we eigenlijk niet zouden willen maken….En nouja dan moet je denken aan dingen als, dat we geen tijd nemen nu voor "code reviews", dat zou je eigenlijk wel willen. Dan moet je denken aan, dat wij hebben op 1 plek code gedupliceerd, terwijl we dat eigenlijk niet zouden willen. Nouja, dat soort dingen."*

Respondent 4 indicates that it depends on how you deal with situations when the customer asks something from you. If you experience that as pressure or not: *"Maar het is een beetje afhankelijk*

van het team waar je in zit, of je wel of niet de druk voelt. Weet je, die klant, die wil natuurlijk iedere keer meer. En die merkt op een gegeven moment ook wel eens, althans binnen XXXXXXX althans wel. Dat er ook wel eens iemand gewoon buiten de product owner om, probeert bij een teamlid, "Joh dit moet nog eigenlijk even mee, lukt dat?" En ik denk dat het ook wel een beetje afhankelijk is van hoe jij als teamlid in elkaar steekt, of je dat als druk ervaart, of dat je denkt van: "Nou weet je, ga daar maar naar toe" en als jij zelf gewoon aangeeft, "dit is leuk, maar dat is niet de manier waarop wij werken hier, als je wat in wilt schieten, dan zet je hem maar op de backlog""

Respondent 1 indicates that pressure is caused by deadlines:

"Dan denk ik aan het team waar ik hiervoor zat, "PVS" dat was de vervanging van de hele XXXXXXX, dat is ook wel in de krant geweest enzo he van "XXXXXXX gaat een maand dicht, want ze moeten een paginaatje vervangen!" Dat moest in een bepaalde tijd af, dat heb ik wel meegemaakt"

Respondent 1 also indicates that pressure is caused by interference of the customer:

"En die interference dat is wel leuk, dat mensen die bedenken iets en dan uiteindelijk dan ben je bijna klaar en dan komt er ergens nog iemand aan en die zegt "Ja, maar hoe zit het dan als het zo is? Oh dat werkt dan helemaal niet. Maar het zou wel deze sprint af zijn." Dus dan wordt er verwacht dat er hard doorgewerkt wordt."

Respondent 1 also indicates that the pressure they are experiencing is caused by factors beyond their control like other teams they depend on:

"Ik voel bijvoorbeeld nu wel heel veel druk komen, want je wilt iets afronden, eind van het jaar moet het klaar zijn. Maar de druk komt eigenlijk meer van factoren waar wij geen invloed op hebben, dus dat zijn andere teams, die wat voor ons moeten opleveren, of wij zijn verantwoordelijk, maar ik mis eigenlijk gewoon iemand die in de gaten houdt, van oke welke stappen moeten we nu doen, om naar productie te gaan. Wat moet er allemaal geregeld worden? Dat missen wij eigenlijk."

Weights

Table 9.19 - Pressure - weights

	Respondent 1	Respondent 2	Respondent 3	Respondent 4	Respondent 5	Respondent 6
Weights	3	-	4	3	2	4

The reason why respondent 6 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that if the team doesn't experience pressure, they can deal with it in a relaxed way, which increases the risk of not being able to deliver on time:

"Ja, dat hangt er vanaf, ik vind het wel belangrijk en zeker in de tijd waarin we nu leven. Ik vind het goed, dat het team ook ervaart dat die druk er is. Ik heb ook wel in een tijd geleefd dat het team die druk niet ervaarde. … En als zij daar dan relaxed mee omgaan, dan heb je dus het risico dat je niet op tijd kan leveren. En dan faal je als team. Dus in die zin vind ik dat wel belangrijk."

The reason why respondent 4 thinks this CSF scores a 3 and is therefore seen as relatively important is because he/she indicates that he/she does not let himself/herself be under pressure so quickly:

“Ik vind het niet belangrijk, ik laat me niet zo snel onder druk zetten.”

The reason why respondent 3 thinks this CSF scores a 4 and is therefore seen as important is because he/she indicates that when the pressure increases, you make more mistakes, so the pressure has to be managed:

“Als je ziet dat de druk heel erg toeneemt...ze zeggen, dat onder druk wordt alles vloeibaar, maar onder druk maak je ook meer fouten. Dus gaat ook je kwaliteit naar beneden. Dus die druk moet je wel degelijk managen.”

The reason why respondent 1 thinks this CSF scores a 3 and is therefore seen as relatively important is because he/she indicates that pressure makes sure there's no loafing:

“Ik ben bang dat er anders teveel gelanterfantert wordt en dat er dan als er niet een beetje druk is dan kom je er denk ik ook niet, want anders krijg je van dat kan later ook wel..., of dat zien we nog wel..”

Performance improvements

No performance improvements were observed.

Measurement of the CSF

Respondent 1 indicates that maybe pressure could be measured in story points, but it remains a bit of a guess:

“Ik zit even te denken of je de story points volgens de scrum. Of je je doel van je sprint gehaald hebt en dat je dan als je dat altijd haalt. Dan kun je zeggen we gaan gewoon meer doen en zo kun je die druk ook weer een beetje opvoeren en dat wordt …. Wij houden die punten niet echt bij van hoeveel hebben we gedaan van hoeveel hadden we te weinig of wat dan ook. Het blijft een beetje nattevingerwerk.”

Because respondent 1 only provides an indication, criteria 3 is not proven.

Implementation of the CSF

Due to a lack of time, the interviewer did not have enough time to ask about the implementation of this CSF.

Conclusion

This potential CSF could not be seen as verified CSF, because the CSF meets only criteria 1.

Appendix ZB

Description of the case organization

The case organization provides ICT services to the business to ensure that business processes can be carried out as (as efficiently/effectively as possible). They automate business processes in the broadest sense, in a slightly narrower sense they provide customized solutions for the implementation of the XXXXXXX.

They have several government parties that are clients for the delivery of custom applications. The implementation of the legislation is generally realised by means of customised packages (self-built software). In addition, they also have the "infrastructure" department for the supply of all hardware and associated software such as workstations, storage, access systems and the facilitation of a data centre.

The case organization has about 3000 employees, of which between 80-85 % are internal- and 15-20 % external employees.

Based on this information we can conclude that the case organization meets the first and second selection criteria. The case organization provides IT services to (end) users and focuses on software production and has an organization size of > 1000 employees.

DevOps context

In the case organization, the definition for DevOps given by the gatekeeper is as follows:

"*That you have the responsibility for creating, managing and also taking applications out of production, which means that "developers" and "operations" - employees must be able to do it together. So that also means that a team with only employees "operations" is not DevOps. You can never have the responsibilities in one team if you have the responsibility for people in multiple teams and the responsibility of people.*"

This definition given by the gatekeeper is also inline with the definition of DevOps given by Erich (2019) we adopt in this study: "interaction between development and operations".

Inside the case organization there are between 45 and 50 DevOps teams assigned. There are also teams that are split up and call themselves virtual 2 teams, which depends on the context from which you look at this. In general, the number of members of a DevOps team consists of 6 to 10 people. There are a few exceptions, the larger DevOps teams who are on the verge of splitting up or the DevOps teams who are under development.

Within the case organization there are the following types of teams:

- Software development teams (DevOps teams)
- Service teams
- Business Analysis Teams (BAT)

The role composition of a service team depends on the goal of the team and can vary from team to team. For example the team: "Test Automation" which only includes people with the "test" role. Just like for example the team: "Integration", which includes more programmers and administrators then the other roles, so depending on the purpose of the service teams, the staffing can be different.

Service teams are primarily intended for creating teams and breaking down teams. In fact, within the case organization it is said that a service team should make itself useless. A service team must ensure that the tooling and the knowledge is stored and will be disseminated in this way, that at some point there can be said goodbye to such a service team. So they are more volatile than the Business Analysis teams and the DevOps teams. Service teams generally contain specialists. Service teams are created at a time when a lot of specialist knowledge is needed, that it is actually impossible to add that specialism to every team.

DevOps teams have a software development process as a basis and delivering and managing applications. Which according to figure 6 can be seen as the “Continuous Delivery pipeline” which includes Continuous Integration, Continuous Delivery and/or Continuous Deployment.

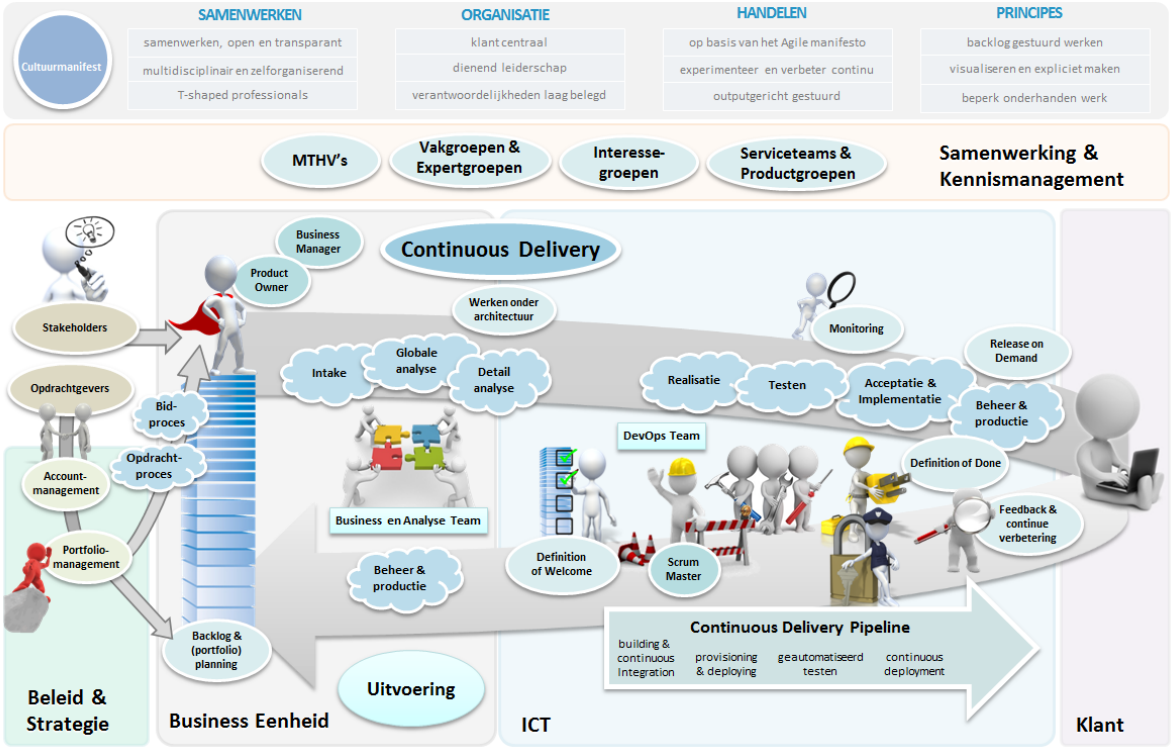

Figure 6 - Context case organization

The DevOps teams have the business as client while the service teams and the business analysis teams have the DevOps teams as client.

Within the case organization there are business analysis teams with many more functional administrators. They are less involved in the process of delivering applications but more in prioritizing the backlog and cutting large parts of the work into small functional chunks. They provide the analysis for the actual DevOps teams.

An average DevOps team has the following 4 traditional roles: "Designers", "Programmers", "Testers" and "Administrators". There is also a role that circles around the DevOps teams, a specific positioning, a "Product Owner" who is not part of the DevOps team. The "Product Owner" is placed outside the DevOps team, but is linked to it. Finally, there are roles that are not necessarily linked full-time to the team, for example: "User Experience Designers" who are involved in screen development and "Security experts".

Based on this information we can conclude that the case organization also meets the third selection criteria. The case organization has established (a) DevOps team(s) which consists of software development team(s) and operation team(s).

Implementation of Continuous Integration, Continuous Delivery and/or Deployment

According to the gatekeeper the objectives underlying both DevOps and Continuous Delivery / Continuous Deployment within the case organization can be found in: "Efficiency", "Agility" and "Being able to deliver faster than the customer expects from us".

Within the case organization Continuous Integration has been their first step, they have put a green check mark behind it, because in general most JAVA based DevOps teams have finished their complete integration street. Some teams (2, or 3 or 4) don't have the tooling at this moment because the investment in the tooling is so big, they can't do Continuous Integration yet.

They do not yet have Continuous Deployment. They don't bring anything automated without the intervention of people to production. This is a business consideration, it has to do with security precautions, they can, but they are not yet at the level that they can build the complete pipeline. However, there is also less added value for them, because they have no commercial advantage in this. They also indicate: "The example that is always mentioned with Continuous Deployment, are companies that have to respond to changes in the market, for example delivering a T-Shirt to the e-commerce shop that wants to deliver T-Shirts within 5 minutes at the moment something is in the news, then Continuous Deployment offers a lot of added value".

They indicate that Continuous Delivery is on the same level as Continuous Integration, Continuous Integration was commonplace 5/6/7 years ago and Continuous Delivery for them over the past 2-3 years has actually reached the level that they can say they are ready in every team.

Based on this information we can conclude that the case organization also meets the fourth selection criteria. The case organization has implemented Continuous Integration, Continuous Delivery and/or Continuous Deployment with a minimum of 2 - 3 years ago.

Continuous Delivery pipeline (process steps) & Roles / T-profiles

Within the case organization, the Continuous Delivery pipeline consists of the following process steps:

They have a source-control system. In the source-control system, projects can be created in which source code can land that has been created by the programmers and testers. (Example: "Git" or "SVN", SVN is still used in some cases.) The developer or tester checks code in, in the source-control system. A project starts with a trigger as soon as code is checking-in into the source-control system.

Then they have the build tooling. In their case it is "Jenkins", they have a complete pipeline configured in there. When that trigger goes off, a Jenkins build pipeline is started, which will then build the application.

Subsequently, the Jenkins build pipeline will perform the various test phases, unit tests, especially the construction tests, which depend on what the configuration looks like, e.g. pulling a test environment in the air and performing functional and non-functional tests. One of the important additions they have made to enable Continuous Deployment is that security tests are conducted. They perform about three security tests to ensure that they can give an indication that the code they have created is safe enough to go into production. In addition, performance tests can also be part of the pipeline.

After that comes the "package" phase, in which the package that has been created is made ready for production and provided with a version number.

As soon as that package is created, they place that package in a library, in their case XL-Deploy, in which they indicate that this package is ready to be executed in any given environment. (Ready to go to production). This "deploy" phase is optional, you can also release the package to production immediately after packaging.

The moment all conditions are met, the package is put in XL-Release, inside Xebia's tool, which is a repository from which you can go to production. In XL-Release you can put all kinds of workflows to specify where a package needs to be deployed to. XL-Release is considered to be in-production.

* The gatekeeper does mention that the "deploy" phase is a big step for the case organization, because: "Imagine that you have a complex set of applications that you want to bring to production together, you might want to run everything again in an acceptation environment to see if it matches. If you go to 1 application, that step is unnecessary, but if, for example, you want to roll things out for the law of XXXXXXX (one of our large systems), you are going to roll out things of 6 teams, so you will have to take one more step to do a test".

This latter indicates that after the "deploy" phase an additional "test" phase might take place. As an example, the gatekeeper mentions: "For example, the connections with the outside world. At some point, you want them to show that they still work and that you have the same versioning, which is actually optional. But it is becoming more and more important because we are working more and more in chains".

Within the case organization, the distribution of roles across the Continuous Delivery pipeline process steps is as follows:

The "Source-control" phase is primarily done by developers (programmers) and testers.

The "Build" phase is in fact fully automated, so if they would do full Continuous Deployment, there wouldn't be any human involved after that.

Only from the package phase, that happens in Jenkins, the deploy phase and eventually the testing, there could still be a human intervention, that might be a tester.

In addition, the gatekeeper indicates that in practice the functions are somewhat more diffuse, because of the T-shape and you could say that those "package", "deploy" and "release" phases are work for the "administrators" in the team, only they won't do it manually, they will configure some things in Jenkins for that. But in practice, you see that developers play a role in these things.

The gatekeeper gives another example concerning the T-shaped roles: "In a not so long time ago, 2 or 3 years ago, there were a lot of test coordinators or test managers running around, who actually did nothing more than cut up complex tests into small chunks so testers could use them, we don't have those functions anymore. In fact, we didn't even acknowledge them as a role anymore. But we do expect that someone on the team, at the point of testing a very complex application, that someone on the team should be able to take over the role of test manager or test coordinator, whatever you want to call it. But that can quite well be a developer or a functional designer".

Another example: "You see functional designers coming in in the morning and switch on the monitor screens, which are also concerned about whether they see anything in red. You could say, "No, that's work for an Ops guy. No, but that designer is indoors anyway and he knows about Logging and Monitoring, so he will go after it. So it gets more diffuse." The gatekeeper calls this: "True DevOps!" namely "getting the responsibility, but primarily taking the responsibility".

Finally, the gatekeeper indicates that within the case organization in practice, the traditional role of the "functional designer" is disappearing more and more into the background. Entering into a conversation with the customer, a functional designer can do that, but this can also be done by a programmer, or a tester. So the role of the functional designer is shifting, just like the role of the administrator. As an example, the gatekeeper quotes: "An administrator and a tester have a lot in common. Because a tester and an administrator both suffer when a programmer makes an error that is not removed. The tester on the day after, and the administrator on the weekend after. So you could also let the role of the tester and the administrator blend together a lot more. So the roles/specialty aren't so clearly to pinpoint on the function you have within a DevOps team anymore".

Some statements about their T-shaped role provided by the respondents:

Some respondents indicated that they were barely or not at all T-shaped. They say that the case organization strives to achieve T-shaped roles and it is being expected to operate T-shaped.

Another respondent indicated that if they have not much to do and somebody else is very busy, to help that person, so they are working in a different discipline. For example, doing research among students, that is something a User Experience person actually does within their team, but from a different discipline.

A respondent has even indicated that he doesn't really want to call it T-shaped, because programming is really his main function, and of course he also tests his own software with unit tests. But they also have their own testers in the DevOps team, who do the functional tests and the "Ops", who eventually take things to production. He also indicates that they are not even allowed to do that.

Another respondent indicates that his/her function as senior programmer / technical designer is the 3rd function level. According to him/her there are multiple function levels: "you've got basic programmer, median programmer, senior programmer, technical designer, and at that level you're supposed to be T-shaped." According to this respondent, T-shaped means that you are specialized in a certain field and that you can oversee the entire software-development discipline and act in it (full-stack). With full-stack he/she means the frontend and the backend.

Based on this information we can conclude that the case organization also meets the fifth selection criteria. The Continuous Integration, Continuous Delivery and/or Continuous Deployment pipeline process steps, role distribution and responsibilities could be clearly explained.

Shared goals

Shared goals vary from team to team. According to the gatekeeper, this is a DevOps effect, which means you don't prescribe what you should do as a team. But on a more abstract level, you tell the teams which goals they have to pursue.

The gatekeeper does indicate that it is a struggle when they are looking at a sprint, to what extent they can influence what happens in the team. So they have to set frames. These frameworks are defined in the "(case organization) way of working" and that is a list of frameworks, rules and guidelines that they can give to the teams.

The teams themselves can also traditionally have all kinds of wishes, they have their own roadmap, their own vision on what they want, for example which techniques they want to use to make things, how they want to do the architecture, so this is partly determined outside the team by their "basic architecture". But teams have a lot of freedom in choosing, and that is also part of DevOps, according to the gatekeeper.

According to the gatekeeper, DevOps is about the roles in the team, the responsibilities in the team and the tooling. Within your DevOps team, you have to be able to make the choice to work with the tool you want as a team. But there is another challenge for the case organization, because if a team wants to work with a very exotic tool, they have a problem because if they have to recruit people for that, they suddenly have to put a very exotic tool on their list of specialties. So for the case organization it is important to inform the teams that they have to use a standard set. And that is where the management of the company comes in, versus the freedom you give to a DevOps team. So if you do want to introduce something new, there is enough space to do this, but the case organization wants that you are going to make sure that the other teams will going to use this as well.
An example of a team with a clear goal is the service team: "Test automation", whose goal is to create an implementation of test tooling for teams that is so simple that testers without specialist knowledge can use it. In summary: "making it easy to implement test tooling in order to automate testing". And there is a "How" above, namely that test automation is becoming obvious. The gatekeeper gives the following example: "Make a fat sausage and present this to the teams so you won't have to force it, teams will be seduced!" The goal you will achieve is that you need less expertise in teams to do the things that are required to make Continuous Deployment a success. This service team is doing test-automation to make Continuous Deployment possible. As an example, the gatekeeper indicates: "Because you don't want to wait until a tester has been called back to work from home on Friday afternoons, because something still needs to be tested. You want teams to be able to run all their test scripts automatically, for that you need knowledge and expertise, which is not necessarily present with every tester. But that's how you make it as simple as possible to do it. That's the goal for this team."

Finally, the gatekeeper indicates that he is not exactly aware internally of the teams what their goal is. He indicates that it is better to ask the respondents themselves about the goal of the team they

are part of. He assumes that a mature team sets its own goals. They have the IV strategy, that is the strategy published by the CIO-office, as ICT they give a turn to that, those are goals they give to teams. But within that, the teams are free to define their own goals.

Some examples of different types of teams with a shared goal provided by the respondents:

DevOps team: “Teachers” or “ABL”

The goal of this team is to rebuild the current ABL application. And the reason for that is that the current ABL application is rather outdated in terms of technology. And secondly, there are several teacher regulations, in addition to the teacher scholarship. And those regulations are now stored in Access applications. They have set themselves the goal, to realize a generic application, in which they can also implement other regulations.

DevOps team: “Site of (case organization)”

The goal of this team is to produce the software for the site of the case organization and to make the site possible.

DevOps team: "Business Services"

The goal of this team is to satisfy the business customers of the case organization, and also to give them the opportunity to manage the funding information and also to be able to give grades to schools, so that schools get their funds. They keep track of the systems which contain information about how many students are enrolled. They also make software, at the moment they are building a completely new application which should eventually offer the possibility to submit ideas for a new school.

DevOps team: “Institutions and education register”

The goal of the entire team is to create and manage the register for institutions and education. This is a register in which all courses and institutions that offer courses are registered. Given the size of the entire team, they have split the team into 2 development teams. One of the development teams aims to generate machine interfaces (Application Programming Interfaces (APIs)) of these. This means that institutions can automatically deliver data, but also automatically retrieve information from a machine interface.

Service team: “Continuous Delivery-team”

The goal of the CD team is to take care of all the tooling needed for software development. The boundary is drawn at the point where it will become part of an application, so if a team uses a framework in the code, (code-based), it does not apply to them anymore, that is for the account of the DevOps team. But everything that is supporting the software development process, that is what the CD team is involved with. In addition, they also take care of the provisioning of the platform (on the virtual layer). So the provisioning of all servers on/in all environments for the applications of the case organization.

Based on this information we can conclude that the case organization also meets the sixth selection criteria. The case organization has established shared goals concerning Continuous Integration, Continuous Delivery and/or Continuous Deployment

Performance requirements

The gatekeeper indicates that there are no underlying requirements, but can imagine that a very mature team makes requirements for themselves.

An example of this is as support for Continuous Delivery, they went to self-service of their environments (self-service provisioning.) So if a team wanted to bring an application on the air they first had to request it and that request took 6 weeks. That was very acceptable at that time, but at a certain point, especially from people who had experience outside the case organization that it could also be faster, who were setting the bar high within the case organization, who said: *"we want to go from 6 weeks to 6 minutes"*. So they went from 6 weeks to 6 minutes in their deployment model because of self-service.

According to the gatekeeper, a mature team often makes requirements for themselves. An example of a requirement given by the gatekeeper: *"Everything we do must be returned within seconds when the runtime is requested"*.

The product-owner of a team often does not impose any performance requirements, unless it was based on an incident or at the moment of dysfunction, it may come as a requirement.

The gatekeeper also indicates that they think DevOps also means that they make fewer and fewer requirements to the teams. But that the teams themselves should come up with those requirements, or should decide for themselves. As an example, the gatekeeper states: *"If you really want to give them the confidence, you have to interfere as little as possible and let it happen along the way. And to let the team be called out of their bed at night, and to let teams work until 6 a.m., it's their problem, it's not the management's problem, they decided not to make any performance requirements... It's their problem."* It can be derived from this that the responsibility to make any performance requirements lies entirely with the team.

Some performance requirements confirmed by more than one respondent:

Performance requirement - Response time

2 respondents indicated that there are technical performance requirements such as a maximum response time of 1 or 2 seconds for the final product.

Performance requirement - Achieve Sprint

One respondent indicated that teams are doing sprint planning, in which an estimate is made of the number of story points, because as a team, the performance requirement is that they actually want to achieve all the estimated story points in a sprint.

For each sprint, each team checks how many story points they have achieved. Then, at a certain point, they get an idea of what they can achieve. Example: "We can achieve X number of story points per sprint with this team".

Another respondent indicated that: *"if you're just starting a DevOps team, you don't know anything yet. So you have to learn this along the way. So you first estimate in points, for example you have an estimate of 50 points. Then you do one of those sprints, and then you realize 20, well, something's not right. Either you estimate too high, or you're too optimistic, and then at some point, over time, you get a picture of what your performance is."* In addition, the respondent indicated that another handy

way to estimate story points at the beginning of each sprint, is to use reference tasks that are equal to a certain number of points.

Measurement of Achieve Sprint - Velocity

According to the gatekeeper, the performance of a team is expressed in Velocity: "X story points per sprint". The teams know what their velocity is. But this is an incomparable value, because each team has its own velocity indicator. In addition, the gatekeeper says that teams often plan in velocity, by taking into account the number of points they can do per sprint. This does not say anything about the number of hours, but it does say something about the amount of work that can be done within a sprint. In general, the teams take a kind of measurement application for this, of average weight, of which they know how much time something takes. They set the velocity or weight of the application to 1, or 100, and 100 is what they can burn in a sprint. (reference task) For example: "Our velocity = 100, so we can build or modify this application in 1 sprint". So they could do a similar application in the same amount of time, in one sprint. This allows teams to measure their progress and see if they have become better, or less, in the sense of realizing more story points (faster), or less story points (slower).

Based on this information we can conclude that the case organization also meets the seventh selection criteria. The case organization has established performance requirements.

Willing to submit interview candidates

The gatekeeper has proposed 6 interview candidates. Based on this information we can conclude that the case organization also meets the eighth selection criteria. The case organization proposed a minimum of 8 respondents to conduct interviews (by a margin/bandwidth of 2 respondents).

Conclusion

Because the case organization meets all selection criteria, this case organization was selected to conduct this research. The gatekeeper was also requested to provide the document that shows the target situation of the context of the case organization with regard to DevOps, Continuous Integration, Continuous Delivery and/or Continuous Deployment. This can be seen in Figure 6.